

UNIX™ REVIEW

THE PUBLICATION FOR THE UNIX™ COMMUNITY

September 1984 \$3.95



UNIX
RELATIONAL
DATABASE
MANAGEMENT
SYSTEM

UNIFY
CORPORATION

UNIX RELATIONAL
DATABASE
MANAGEMENT
SYSTEM

REFERENCE
MANUAL

UNIX RELATIONAL
DATABASE
MANAGEMENT
SYSTEM

TUTORIAL
MANUAL



UNIFY FINISHES FIRST!

In one independent competition after another, UNIFY has proved itself the fastest UNIX data base management system. No wonder it's been selected by more computer manufacturers than all other UNIX data bases combined.

UNIFY speeds you through development and expedites program execution with some of the most powerful utilities of all, including:

Fully menu-driven design.

A powerful screen handling package that helps you format screens quickly, with no coding required.

Raw I/O, that lets you bypass the UNIX file system for up to 40% faster performance in large data bases.

Built-in optimizers that select the fastest of four data access methods.

Industry standard IBM SQL query language, plus our powerful report

writer, for easy access by end-users.

Ninety subroutines for advanced program development . . . the most complete package of its kind.

UNIFY's integrated design links program modules like screens, query language and report writer to help you quickly create complete, friendly, easily expandable applications.

Horsepower for the long run. Unlike other data bases, UNIFY won't slow down under the weight of additional data or multiple users. It's built with the power to support new features later.

Judge for yourself. Send for our 300-page tutorial and 500-page reference manual — yours for only \$95 — that show you how to build virtually any application. Contact UNIFY, Dept. UR-9, 9570 S.W. Barbur Blvd., Portland, OR 97219, 503/245-6585.

UNIFY[®]
THE PREFERRED UNIX DBMS.



UNIXTM REVIEW

THE PUBLICATION FOR THE UNIX COMMUNITY

Volume 2,

Number 6

September 1984

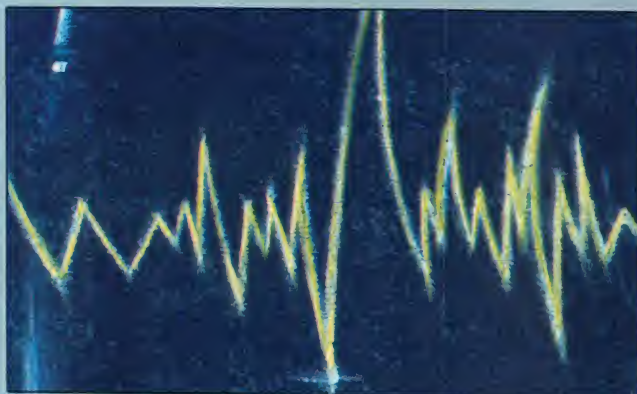
DEPARTMENTS:

- 4** **Viewpoint**
- 8** **Devil's Advocate**
by Stan Kelly-Bootle
- 12** **The Human Factor**
by Richard Morin
- 18** **C Advisor**
by Bill Tuthill
- 68** **Industry Insider**
by Mark Sobell
- 72** **Rules of the Game**
by Glenn Groenewold
- 78** **/usr/lib**
by Jim Joyce
- 88** **The UNIX Glossary**
by Steve Rosenthal
- 104** **Calendar**
- 108** **Advertiser's Index**

The cover illustration was created on the UC Berkeley UNIGRAPHIX system, designed by Professor Carlo H. Sequin and a group of his graduate students. Dubbed the "Granny-knot Lattice," the graphic was produced in 20 minutes of computer time. Color was added by airbrush artist Hyon Kim.

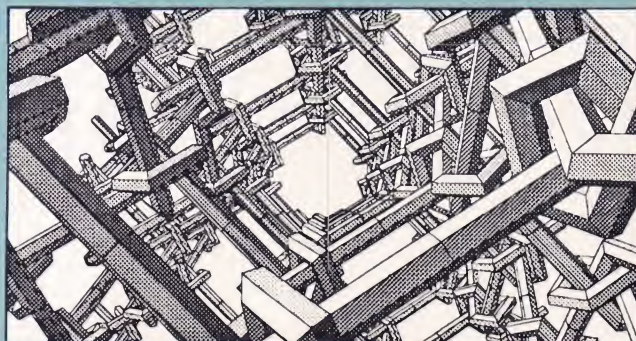
FEATURES:

- 26** **KEEPING YOUR SYSTEM HEALTHY**
by Jim Joyce and Douglas Merritt
A recommended regimen for keeping systems fit.

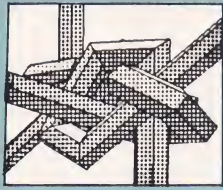


Tips for keeping your system running.

- 38** **LINKING UP WITH THE OUTSIDE WORLD**
by Bob Toxen
An introduction to UUCP.



Picking through the **UUCP** maze.



SYSTEM ADMINISTRATION

42 **TERMCAP UNVEILED**

by Douglas Merritt

Pages out of a wizard's handbook.



That old black **termcap** magic.

52 **SECURITY ROUNDTABLE**

by Dick Karpinski

Interviews with leading experts in the field.



Ed Gould opens the *UNIX REVIEW* security talks.



Chilling system administration tales.

62 **HORROR STORIES**

by Dr. Greg Chesson

Tales of system administration woe suitable for late-night reading.

UNIX REVIEW (ISSN 0742:3136) is published monthly by REVIEW Publications Co. It is a publication dedicated exclusively to the needs of the UNIX community. Subscriptions are \$28.00 per year (12 issues). Canada and Mexico add \$7/year. Overseas add \$20/year for surface mail. Address correspondence regarding editorial, press releases, product announcements to 570 Waller Street, San Francisco, CA 94117. Correspondence regarding subscriptions, change of address, USPS Form

2579, fulfillment and dealer sales should be addressed to 901 S. 3rd Street, Renton, WA 98055. Letters to UNIX REVIEW or its editors become the property of the magazine and are assumed intended for publication and may so be used. They should include the writer's full name, address, and home telephone. Letters may be edited for the purpose of clarity or space. Opinions expressed by the authors are not necessarily those of UNIX REVIEW. Entire contents copyright © 1984. All rights reserved and nothing may be reproduced in whole or in part without prior written permission from UNIX REVIEW. Editorial telephone: 415/621-6415. Subscription/Dealer Sales telephone: 206/271-9605. UNIX is the trademark of Bell Laboratories, Inc. UNIX REVIEW is not affiliated with Bell Laboratories.

EDITOR:

Mark Compton

MANAGING EDITOR:

Pamela J. McKee

ASSOCIATE EDITORS:

Ken Roberts

Scott Robin

PRODUCTION DIRECTOR:

Nancy Jorgensen

PRODUCTION STAFF:

James Allen

Dan Arthur

Tom Burrill

Cynthia Grant

Corey Nelson

Florence O'Brien

Denise Wertzler

CIRCULATION

Tracey McKee

Barbara Perry

BUSINESS MANAGER:

Ron King

ADVERTISING**REPRESENTATIVES:**

Jules E. Thompson, Inc.

1290 Howard Avenue Suite 303

Burlingame, CA 94010

Lucille Dennis-415/348-8222

303/595-9299 - Colorado

312/726-6047 - Illinois

617/720-1888 - Massachusetts

713/731-2605 - Texas

Jules E. Thompson, Inc.

2560 Via Tejon

Palos Verdes Estates, CA 90274

Mary Winchell-213/378-8361

212/772-0933 - New York

TYPESETTING:

Data & Staff Service Company

PRINTING:

Thomas Ogle

EDITORIAL ADVISORS:

Dr. Stephen R. Bourne Software

Engineering Manager, Silicon

Graphics, Inc.

Jim Joyce, President, International

Technical Seminars, Inc.

REVIEW BOARD:Dr. Greg Chesson, Technical Staff,
Silicon Graphics, Inc.Ted Dolotta, Senior Vice President of
Technology, Interactive Systems
CorporationGene Dronek, Director of Software,
Aim TechnologiesDavid Fiedler, President, InfoPro
SystemsGeorge Goble, Systems Engineer,
Purdue UniversityBill Joy, Vice President of Research
and Development, Sun
MicrosystemsJohn Mashey, Software Engineering
Manager, Convergent
TechnologiesRobert Mitze, Department Head,
UNIX Computing System
Development, AT&T Bell
LaboratoriesDeborah Scherrer, Computer Scientist,
Mt. XinuJeff Schriebman, President, UniSoft
CorporationOtis Wilson, Manager, Software Sales
and Marketing, AT&T Technology
SystemsWalter Zintz, Executive Director,
Uni-Ops**CONTRIBUTING EDITOR:**Ned Peirce, Systems Analyst, Bell
Laboratories

VIEWPOINT

Before running into UNIX, I thought that people who argued, "Small is beautiful," were advocates of simplicity. Perhaps they were, but at least within the context of UNIX, I can now see that smallness and simplicity are not necessarily bedfellows.

A convincing argument could be made suggesting that users of small UNIX systems have it tougher than people who do their work on minis or mainframes. The difference is system administration, the theme of this issue.

Two years ago, there wouldn't have been a big enough body of readers to merit an issue devoted to system administration. At that time, after all, system management was the province of the wizard at the end of the hall — the Keeper of the Manuals known as the System Administrator. Whenever problems arose, one simply dropped them at the wizard's doorstep, confident that sooner or later, they would magically disappear. What could be simpler?

The advent of small machines capable of supporting from two to four users (regardless of what the advertisements say) and a growing armada of single-user systems (like IBM's PC/IX), is quickly eroding such simplicity, however. By default, we all now must don the System Administrator's robes from time to time to attempt our own incantations and imprecations.

This wouldn't be so bad if there were a decent script to follow, but the mournful state of UNIX documentation has kept system administration something of a black art.

In this issue, we try to demystify some key points. Jim Joyce and Doug Merritt draw

on their considerable system administration experience to prescribe a number of practices for keeping a system healthy, sprinkling in helpful shell scripts for good measure. Merritt goes on to probe the mysteries of **termcap** in yet another article. **UUCP**, the bane of many a system administrator, is discussed in a third article by Bob Toxen, an acknowledged guru. Dick Karpinski, the manager of UNIX services at UC San Francisco, caps the theme with the first of a two-part series of in-depth interviews with leading UNIX security experts. This month, Bob Morris of AT&T Bell Laboratories and Ed Gould of Mt. Xinu have their say. Next month's installment will focus on George Goble of Purdue University and Bob Chancer of AT&T Bell Laboratories.

It's all eye-opening reading, but we stop far short of exhausting the topic. Many incantations are left unexplained and numerous tricks must yet be exposed. The sheer limitations of space force us to make painful choices — about not only the sorts of topics we'll cover but about the kinds of readers we'll speak to. Most of the articles in this issue assume at least some familiarity with the topic at hand. This, sadly, may leave some readers at the starting gates, but without such assumptions, we could never get to the meaty topics that much of our readership demands.

It's for those readers that this issue has been devoted to simplicity. We'll never again see the Golden Age of system administration, when woes could be sent to the end of the hall to die quietly. But we can remove some of the shrouds that obscure the topic. Here's to clarity...

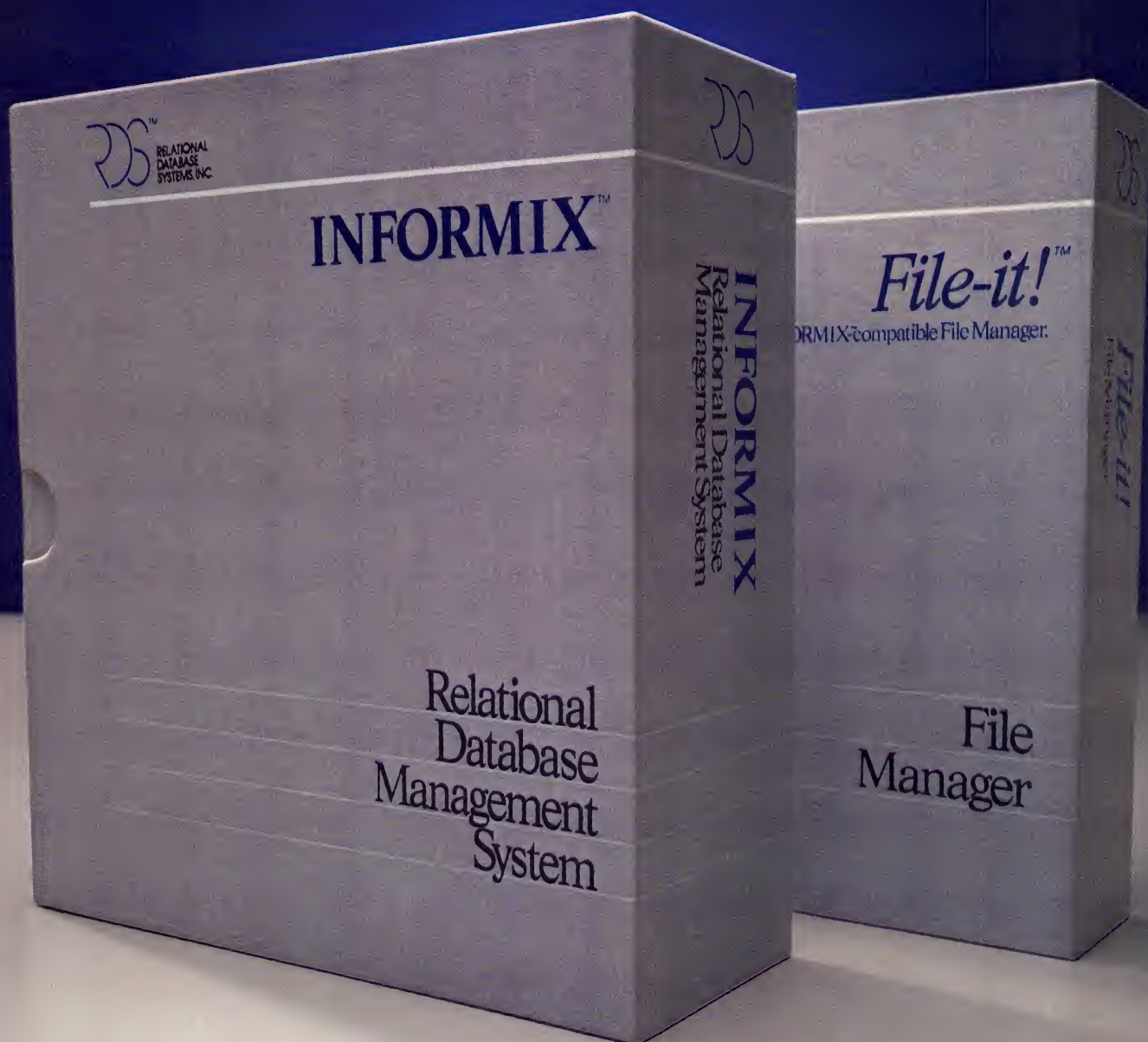
Mark Compton

**YOU CAN'T
PREDICT
THE FUTURE.**

**BUT
YOU CAN BE
PREPARED
FOR IT.**

WITH INFORMIX AND File-it!

THE FIRST DATABASE SOFTWARE FAMILY
FOR UNIX AND MS-DOS.



Now OEMs and systems integrators can sleep better at night. Because one company has taken the worry out of buying the right software.

RDS.

The company that produces a family of database software designed to take on the future.

Incompatibility is a thing of the past.

INFORMIX® and File-it!™ are compatible with UNIX™, MS™-DOS, PC-DOS™, and PC/IX systems (over 60 micros and minis* at last count).

INFORMIX is a true relational database system designed to take full advantage of the power of UNIX. It includes the most widely used report writer on the market.

Then there's File-it! The first easy-to-use UNIX file manager. Together, they have the flexibility to accommodate novices and experts alike.

INFORMIX and File-it! are fully integrated. Users can upgrade from File-it! to INFORMIX or access data from one program or the other without re-entering data, retraining employees or reprogramming.

Applications can also be moved from MS-DOS to UNIX and vice versa without having to rewrite the application.

Simplify program development.

RDS offers C-ISAM™, the de facto standard ISAM for UNIX. It's a library of C subroutines with a B⁺-Tree based access method that stores, retrieves and modifies data from indexed files. It's embedded in INFORMIX and File-it! Or is available as a standalone product.

Software good enough for AT&T.

AT&T, inventor of UNIX, has co-labeled INFORMIX, File-it! and C-ISAM to run on their full AT&T 3B Computer line (from micros to minis).

Hewlett-Packard, Altos, Zilog, Siemens, Cromemco, Perkin-Elmer, Sydis and General Automation have selected RDS as well.

In fact, INFORMIX has an installed base of over 6,000 copies. And RDS has sold over 35,000 licenses for all their products to date.

But before you make up your mind, check the facts one more time.

There's only one database software family that's UNIX-, PC-DOS-, MS-DOS- and PC/IX-based. It runs on more than 60 systems. And it's ideal for both novice and expert.

Now it doesn't matter where the future's headed. You're already there.

*RDS products are available for the following systems:

Altos 586, 986, 8600, 68000	Hewlett-Packard 9000
Apollo DN300	Series 200, 9000 Series 500
AT&T 3B2, 3B5, 3B20,	IBM PC, PC-XT
AT&T Personal Computer	Intel System 86/330
BBN C machine (all models)	Masscomp NC 500
Bunker Ramo Aladdin 20	Momentum Hawk 32
Charles River Data Systems	NCR Tower
Universe 68	Onyx C8002, C8002A
Convergent Technologies	Perkin-Elmer 3210, 3220,
Miniframe and Megaframe	3240, 3250
Corvus Systems Uniplex	Pixel 100/AP, 80 Supermicro
Cromemco System 1	Plexus P/25, P/35, P/40, P/60
DEC 11/23, 11/34, 11/44, 11/60,	Pyramid Technologies 90x
11/70, VAX 11/730, 11/750,	Radio Shack Model 16
11/780	SCI Systems IN/ix
Dual Systems System 83	Wicat System 150WS, 160,
ERG Mini System (all models)	200
Fortune 32:16	Zentec 2020
Forward Technology 320	Zilog System 8000
General Automation Zebra	(all models).
(all models)	

Demos of INFORMIX and File-it! are available. Demonstration software and complete manuals included.



**RELATIONAL
DATABASE
SYSTEMS, INC.**

2471 East Bayshore Road, Suite 600, Palo Alto, California 94303
(415) 424-1300 TELEX 467687

INFORMIX is a registered trademark of Relational Database Systems, Inc. RDS, File-it! and C-ISAM are trademarks of Relational Database Systems, Inc. UNIX is a trademark of AT&T Bell Laboratories. MS is a trademark of Microsoft and PC-DOS is a trademark of International Business Machines.

See us at UNIX SYSTEMS EXPO/84, Los Angeles, Booth #550 and UNIX EXPO,
New York, Booth #123

Circle No. 2 on Inquiry Card

DEVIL'S ADVOCATE

Fun and games

by Stan Kelly-Bootle

I have bad news for all those readers who have been urging me to embellish the action in my reportage of Civil War II. *UNIX REVIEW*, unlike other less scrupulous journals, will never resort to fabrication in order to titillate bored, terminal-bound readers. But if it turns out that IBM is actually using AT&T prisoners of war for bayonet drill, rest assured that my pen will put you right there behind home plate.

In the meantime, to celebrate System Administration Day, the opposing troops have cautiously left their frontline trenches and are fraternizing (or sororizing, as the mood takes them) in no man's land. On this holiest of holy days, all past grudges and conflicts are put aside as makeshift games of volleyball and Donkey Kong blossom along the battlelines. This, traditionally, is the one day of the year when a complete moratorium on all systems changes is declared at all sites.

"No patch, however pressing, may be patched this day. Neither may thy kernel be tweaked by thy kludge" (St. Opel, III: vi). System administrators throughout the land perform their solitary devotions from dawn 'til dusk at deserted sites, rejoicing in the blessed freedom from users, and taking advantage of the rare opportunity of putting the previous



year's amendments into `/usr/man` and peeking at everybody's mail.

The rules governing System Administration Day, like the rules of UNIX, are largely unwritten. A powerful oral tradition stretching back to the first misty stirrings of human consciousness (circa 1969) suffices to maintain a remarkably uniform and portable corpus of dogma and ritual. The master CRT must be aligned toward the June solstice and — as the first rays of the rising sun strike the RETURN key — the invocation:

```
# status__quo__vadis?
```

or mnemonically:

```
# wgy?
```

is entered. The system then accesses `/etc/mantra` and conducts a rigorous, interactive catechism

with the system administrator. I am honor bound not to reveal this hermetic dialogue to the uninitiated, but in the name of courageous journalism and increased circulation, here are a few meaty extracts:

Q: O What'll We Do With the Hog's Eyes?

A: We'll Make Them Into Puddings and Pies!

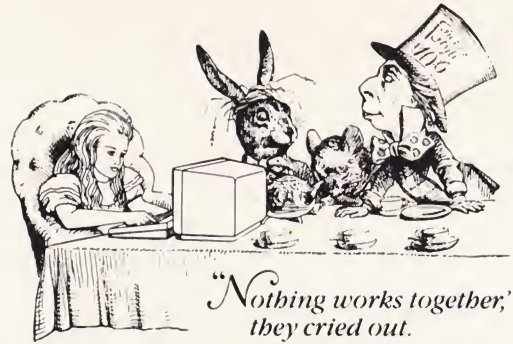
Q: And What'll We Do With the Hog's Cheek?

A: We'll Change His Password Every Week!

(This example assumes 4.2 BSD with the optional iambic-tetrametric-rhyming-couplet feature.)

There then follows a playful "revenge" session during which the **alias** and **history** specifications of unpopular users can be "suitably clobbered." Finally, the system administrator, in one mad mayfly fling, practices the sending of disconcerting messages to remote terminals — an essential skill of the trade where timing is all important lest jobs be aborted at non-critical moments.

During the switch from Julian to Jamesian calendars (see, for example, *Jules ou Jim? Un Calendrier de Nos Jours* by Frederick LeMontre), System Administration Day moved from the nearest Friday to June 22, and fanciful etymologists were quick to point



*"Nothing works together,"
they cried out.*

"It will now," said Alis.

With Alis™ everything works together. Text, spreadsheets, drawings, business graphics and database information work together in a single, always editable document.

Alis combines the advantages of integrated PC applications with the information-sharing benefits of communications-based OA systems. It offers the most advanced total office solution ever.

Alis makes it easy for people to work together. It provides integrated electronic mail, calendar and meeting scheduling, and revolutionary Automatic Office Assistants™ to aid management information monitoring and decision making.

Written in "C," Alis is initially available on UNIX* to large OEMs. It's destined to bring sanity to the mad tea party world of office automation.

*UNIX is a trademark of Bell Laboratories. Applix, Alis, and Automatic Office Assistants are trademarks of Applix, Inc.

Alis™

The next-generation office software system from APPLIX

Finally, some answers in Wonderland.

APPLIX, INC. 302 TURNPIKE ROAD, SOUTHBORO, MASSACHUSETTS 01772 (617) 481-4721

out that after SADay came a SAD-
DERDay. And so it was and is: for
the administrator, a return to the
daily hurly-burly; for the systems
programmer, a whole day's patch-
ing to catch up on; and for the
Hog, a terrible spell cast over his
or her files! By the by, have you
ever tried to use `cs` with the
terminal in `-echo` mode?

So we are left with a nagging
question and an unexplained
mystery. Why do some disastrous
events become hallowed and an-
nual? How is it that during the 24
hours of SADay, every clock-chip
in the world STOPS TICKING?

**The rules governing
System
Administration
Day, like the rules
of UNIX, are
largely unwritten.**

*Stan Kelly-Bootle is a grizzled
mainframer who worked on the
pioneer EDSAC I at Cambridge
University in the early 1950s. As
founder/President of the LISA
Moaners' Club, he urges more
machismo and less user-molly-
coddling in software. In spite of
some reservations, he feels that
UNIX is a bandwagon heading in
the right direction. His exposé of
computer scientific epistemology in
the lexicographic environment, "The
Devil's DP Dictionary" (McGraw-
Hill, 1981), is to appear soon in a
Japanese language edition (Shizen
Sha, Tokyo). ■*

TRAINING

SYSTEM V

For 15 years, we've taught our own people to use
the UNIX™ System. Now we can teach yours.

WHY AT&T FOR UNIX SYSTEM TRAINING?

AT&T offers the most current and comprehen-
sive training on UNIX Systems.

AT&T provides the best learning environment;
one terminal per student; evening access to facil-
ities; and expert instructors.

AT&T has the breadth of courses your staff
needs to unlock the full power of UNIX System V.

AT&T courses signal your commitment to
improving productivity with high-quality training
for your employees.

AT&T COURSES OFFER:

The same training and methods we use to

teach the UNIX System to our own people.

Rigorous classes designed to teach specific
skills for job-specific applications.

Five areas of instruction ranging from intro-
ductory to advanced levels for Managers/Supervi-
sors, Users, Systems Administrators, Applications
Developers, and Systems Programmers.

Frequent class offerings so you won't have to
wait for the courses you want.

Conveniently located training centers
in Princeton, NJ; Columbus, OH; Lisle,
IL; and Sunnyvale, CA. Or we'll bring
our courses to your company and hold
the training at your convenience.

For more information, a
catalogue, or to register for classes,
call 1-800-221-1647, Ext. 87.



CMC COMPUTER METHODS LIMITED



- Interactive Spelling
- Balance Sheet Mathematics
- Auto Footnoting
- Auto Outlining
- Phrase Glossary
- Template Assembly
- Photo Typesetter Interface
- Widow-Orphan Elimination
- Pagination
- Form Generation
- Electronic Mail
- Data Entry
- Financial Analysis

DOCUMENT PROCESSING SYSTEMS

Box 709 Chatsworth, CA 91311 (818) 884-2000 TWX910-494-1716 Int'l Tlx 292-662 XED UR

THE HUMAN FACTOR

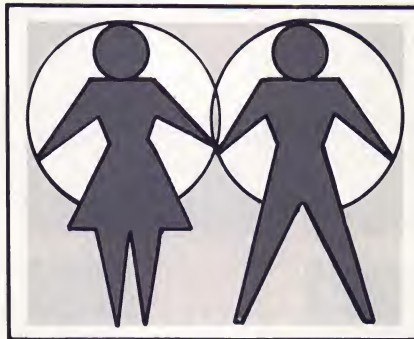
Let's be perfectly clear

by Richard Morin

The phone company ... offers this tolerant message: "We're sorry, but we were unable to complete your call as dialed. Please hang up, check your number or consult your operator for assistance." ... I suspect that a programmer might have generated: "Illegal phone number. Call aborted. Error number 583-2R6.9. Consult your user manual for further information."

Dr. Ben Shneiderman, "System Message Design: Guidelines and Experimental Results; Directions in Human/Computer Interaction." Edited by Albert Badre & Ben Shneiderman. (Ablex Publishing Corp., 1982)

UNIX isn't the only operating system with atrocious error messages. It isn't even the worst of the lot. Still, its messages are bad enough to confuse any novice and cryptic enough to occasionally baffle even the most experienced hacker. Let's examine the diagnostics found in the UNIX environment to see why, using the messages found in *Responses to UNIX Commands*, by Henry McGilton and Rachel Morgan. The standards applied come from the following slightly abbreviated ver-



sion of Dr. Shneiderman's criteria for error messages:

GOOD	BAD
brief	wordy
comprehensible	cryptic
specific	general
positive in tone	negative
constructive	critical

Be Brief and Comprehensible.

Though it is easy to produce error messages that are both wordy and cryptic, it is very difficult to be simultaneously brief and comprehensible. UNIX is all too terse as a rule, and its error messages are no exception. The "mkdir: arg count" message is a representative example, printed when **mkdir** is called without any arguments. Keeping some of the traditional UNIX brevity, let's try for less cryptic messages.

Be Specific. An error message is

more specific when it gives more information about the nature of the problem. The "syntax error near line #" message offered by **awk** is a good (bad) example of an error message that is too general. It gives the user essentially no help in finding the syntax error.

Many UNIX error messages have this deficiency. Consider the examples listed in Figure 1 (on page 14).

Most of these messages give the name of the complaining program, which is critically important. The UNIX "tinker-toy" approach to software development means that an error message can come from any one of a number of programs. The user may have constructed an alias, a command line or a shell script using any number of UNIX commands. The message may even have come from a program running in background or as a daemon. Given the name of the offended program, the user can at least start looking for help on an appropriate manual page.

Only one of these messages fails to print the offending character. This isn't critical, but it can still be a nuisance to have to figure out what a program is complaining about.

In any case, note that the same user mistake is described differently in each of the messages in Figure 1. Perhaps this is intend-



CREATIVE CONNECTIONS

Making creative connections for UNIX developers and UNIX users has been the essence of our business since 1977.

We start with the UNIX system itself, adding functionality, performance, robustness and ease of use to produce our UNITY operating system. We then add programming tools and our CHRONICLE business applications family to provide a powerful environment for UNIX-based applications.

To our superior products, we add a superior commitment to customer satisfaction. We treat our customers as our UNIX technology partners. For all our clients - manufacturers, OEMs, VARs and direct accounts - we provide product plus custom development, UNIX components plus enhancements, training plus the industry's finest support. At HCR, our ultimate product is...

PARTNERSHIPS IN **UNIX**CELLENCE.

Human
Computing
Resources
Corporation



10 St. Mary St., Toronto, Ontario, Canada, M4Y 1P9 Tel. (416) 922-1937 Telex 06-218072

See us at the UNIX SYSTEMS EXPO/84, Los Angeles, Booth #651

See us at UNIX EXPO, New York, October 16-18, Booth #347.

Circle No. 68 on Inquiry Card

UNIX is a trademark of AT&T Bell Laboratories. UNITY and CHRONICLE are trademarks of Human Computing Resources Corporation.

```
col      col: bad option X
grep     grep: unknown flag
mail     mail: unknown option X
make     make: unknown flag argument X
nm       nm: invalid argument -X
prep     Unrecognized flag: X
ptx      Illegal argument: X
sed      Unknown flag: X
```

Figure 1 — Some example error messages.

ed as a cure for the distressingly repetitive nature of diagnostics. In that case, perhaps a random number generator should be used to select an appropriate combination of keywords. Otherwise, we might consider opting for a bit of regularity.

Be Positive and Constructive. UNIX diagnostics rarely give suggestions about possible remedies, and the examples above are unfortunately typical of the tone employed. Experienced hackers are generally immune to these types of hostile and negative messages. Novices, however, are more easily frightened and should perhaps be protected from this sort of abuse. In any case, sufficient information should be given to assist the user in identifying the nature of the problem.

WHAT SHOULD BE DONE?

In the June column, I made some fairly rash assertions, such as:

Cleaning up and standardizing error messages would be a modest but very worthwhile effort for the UNIX industry to tackle. Some minimum standards for format and content could be developed and implemented.

At the risk of appearing immodest (cough), your author would like to share some ideas

about the aforementioned standards. There is nothing terribly novel here, which in the world of standards is good. In addition, no terribly difficult tasks are asked of the C hacker(s) who might have to do the dirty work of implementing these suggestions.

All error messages should contain a description of the offending item. The list in Figure 1 might lead one to believe this is common. Unfortunately, UNIX error messages often fail to print the offending flag, filename or whatever, and seldom give any other useful data.

All error messages should begin with the name of the offending program. Many UNIX diagnostics already do this. The current "name: message..." syntax is fine, but see my next point.

Error message syntax and content need to be predictable. A given type of error should yield consistent diagnostics across the range of commands. Further, the user should be able to tell which fields yield what meanings. An at-

```
chmod    chmod: can't access foo
diff     diff: foo: No such file or directory
grep     foo: No such file or directory
ld       ld: foo: cannot open
lpr      lpr: cannot access foo
rm       rm: foo nonexistent
```

Figure 2 — Different messages, same error.

tempt to access the imaginary file "foo" on my computer yields the diagnostics listed in Figure 2.

A novice would be hard pressed to tell what is going on. The program is in the first field in some cases, the filename is first in others. How about a stock message like:

pgm: File "foo" not in specified directory

The diagnostic above performs its function cleanly and unambiguously. A novice could understand it and an expert would recognize it without even having to read the full text.

We want to include enough information to be useful, but long-winded messages waste time. They take longer for the computer to write, and more time for the user to read. How long should our messages be?

A novice can read an 80-character message in about three seconds. An expert can recognize the same message in perhaps half a second. We can ignore the fact that writing and reading proceed simultaneously, and simply add the writing and reading times together to arrive at the figures displayed here:

baud	write	expert	novice
rate	time	time	time
110	7.3	7.8	10.3
300	2.7	3.2	5.7
1200	0.7	1.2	3.7
9600	0.1	0.6	3.1

A number of things become

Follow The Leader.

There's no substitute for leadership. Especially in a market that's moving as fast as the UNIX™ market.

Four years ago we put millions of dollars into developing a new kind of office automation software: powerful software that could "tame" UNIX. Software designed by a physician for simplicity: easy to learn, easy to use.

The result is the Horizon Software System™—an integrated set of sophisticated office productivity tools for shared-logic systems. The core is Horizon WordProcessing™. It's as functional as any you'll find, for any operating system. It's integrated with the Horizon Spreadsheet™, which offers the

largest electronic worksurface anywhere: 256 cells square. They're supported by clerical shortcuts like spelling correction, mail/merge and sorting. And the Horizon system is even available in French, German and Italian. NOW.

The best part is, they were designed from the start to be integrated, and they were designed from the start for UNIX. The language is C, and that means easy portability. Documentation? It's complete. Ask any Horizon user (there are thousands of them).

Don't risk "Johnny-come-lately" software that's been moved from another operating system. Don't risk software that isn't

proven in the market.

If you want software you can learn in minutes, if you want to join the largest installed base in UNIX, if you want powerful, elegantly simple software right now, then follow the leaders.

To Horizon. The standard for office automation in the UNIX world.



The Standard in UNIX™ Office Automation

Horizon Software Systems Inc., China Basin Building—185 Berry Street, Suite 4821, San Francisco, Calif. 94107 • (415) 543-1199

Horizon WordProcessing, Horizon Spreadsheet, and Horizon Software System are trademarks of Horizon Software Systems, Inc. UNIX is a trademark of AT&T Technology.

clear in this example. On a terminal that operates at 1200 baud or above, neither the novice nor the expert is significantly hampered by the writing time of the message. The reading time of the message is unimportant to the novice, who needs the information. The expert's recognition time is small to begin with, so there is

a certain immunity to message length on that level.

On 110 and 300 baud terminals, brevity is of overwhelming importance, however. The look of current messages is perhaps a reflection of the ASR 33 terminals in use when UNIX was created. If we assume that 1200 baud and above will be generally available

in the future, we can safely consider using messages of up to 80 characters.

Some of the current UNIX commands give "usage" messages. This gives the user immediate mnemonic aid in reformulating commands, but it makes diagnostics more wordy. Dr. Shneiderman suggests that systems be built with multiple levels of help facilities. Two ways of accomplishing this suggest themselves.

A "u" command could be created which would give the usage of a specified command, with the default being the last command entered. This should not rely on the presence of **man** files, since they are too large to be stored comfortably on small systems.

Additionally, a shell variable could control the verbosity of error messages. The default mode could be verbose, giving usage information automatically. Experts could then demand a terse mode, causing the shell to skip over the ancillary information.

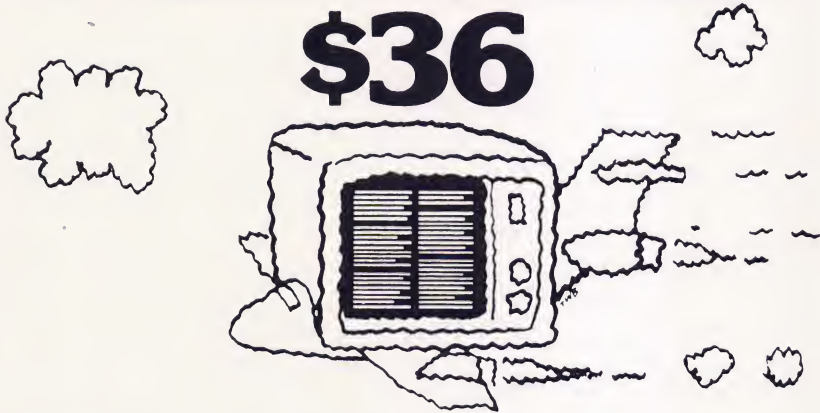
If UNIX is to escape from its current status as a "cult" operating system, it will have to clean up its act. Cleaning up the error message jungle is an obvious first move, and would be very cheap when weighed against its human factors payoff. Standardization is very important in this since a user should be able to expect the same error messages on different UNIX implementations. While AT&T could impose a standard, a better standard might be developed by an organization such as /usr/group. In any case, we gain nothing by waiting.

Richard Morin is an independent computer consultant specializing in the design, development and documentation of software for engineering, scientific and operating systems applications. He currently operates the Canta Forda Computer Lab in Ft. Washington, Maryland.

/USR/TROFF™

ROUND TRIP NEW YORK/ PORTLAND

\$36



From the Big Apple to UNIX™ Paradise and back. 5 minutes out, 48-hour return. TYPESETTING at 15¹/₂ feet-per-minute—wider than this entire UNIX™ Review page (about 1¹/₂" wider).

Telecommunication of documents, including mathematical formulae, gets TYPE SET and RETURNED on first class ACCUCOM-ODATIONS. No waiting at the TYPESETTING counter, and no reservations required. We also "book" passage for magnetic tape and raw text.

Call for details—1-800-ACCUCOM (222-8266)

UNIX is a registered trademark
of BELL LABORATORIES

Circle No. 8 on Inquiry Card



(503) 684-2850

9730 SW Cascade Blvd. / Suite 200 / Tigard, Oregon 97223

WHY DEC AND INTEL CHOSE THE MARK WILLIAMS C-COMPILER.

DEC and INTEL wanted the best C technology available, with excellent code density, supporting the full C language and their specific operating environments—all at a competitive price.

They found it all at Mark Williams.

WHY YOU SHOULD CHOOSE THE MARK WILLIAMS C-COMPILER.

Our C-compiler supports the dominant 16-bit microcomputers—68000, PDP-11, Z8000, 8086—with a proven reliable, high-technology product. We are shipping versions of C for a large number of environments including CP/M and PC DOS. Both cross and native compilers are available.

Call us for the distributor nearest you. OEM's should contact us directly about their specific requirements.

Mark Williams Company,
1430 West Wrightwood, Chicago, Illinois 60614,
312/472-6659.



C ADVISOR

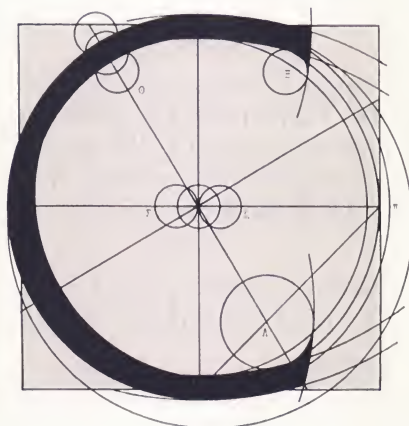
Being and time

by Bill Tuthill

One of the system administrator's responsibilities is to keep system time accurate. When bringing up a UNIX system after a power failure, for example, the superuser needs to reset the time with the **date** command. Most computers cannot keep accurate time, so the system administrator must also reset the clock occasionally even when there has not been a power failure. Some installations, in fact, have hardware that monitors the shortwave radio station WWV, which broadcasts nothing but the precise time. Administrators at most installations phone up for the time or use digital wristwatches, which are usually more accurate at keeping time than most computers.

This article provides a behind-the-scenes look at how UNIX keeps track of time, and contains a few simple programs for gaining access to the system clock. To retrieve the system's notion of date and time, you generally use the **time(2)** system call, in conjunction with **ctime(3)** or **localtime(3)**.

But before discussing these



systems calls, let me deliver the alarming news that time is running out for UNIX. The last tick will occur at 03:14:07 Greenwich Mean Time (GMT) on January 18, 2038. No, this is not the prediction of an apocalyptic religious group. It is pure arithmetic. Since UNIX time is kept in seconds measured from 1970 and is held on most machines in a **long** integer variable, it will become negative on January 18, 2038. Time will then march backwards, as it were. Perhaps this is of little concern, because UNIX will almost certainly be outmoded by the year 2038. But if UNIX *is* still around, alter-

nate arrangements for storing time could be made.

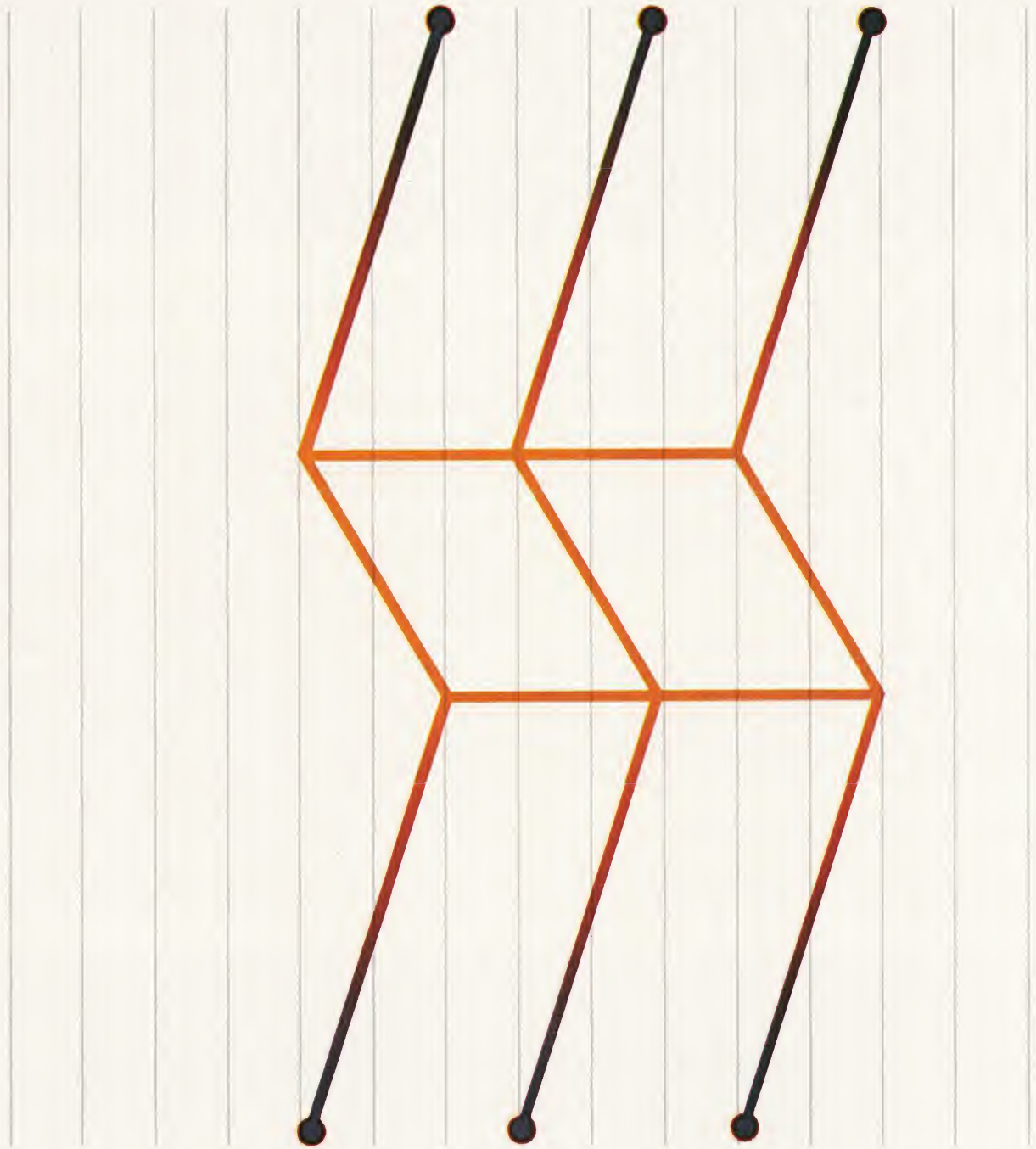
On IBM OS/360 systems, time is stored as a 64-bit quantity and is measured in microseconds from January 1, 1901. Since the maximum 64-bit quantity is much larger than the maximum 32-bit quantity, OS/360 time will last well past the 21st century. Thus, arithmetic alone indicates that IBM will last longer than UNIX.

Many people wonder why the UNIX library routines to get the date and time, such as **ctime(3)** and **localtime(3)**, must be passed the address of a variable containing the time. After all, neither **ctime()** nor **localtime()** modifies the variable passed to it. And the variable isn't likely to change in the milliseconds between the system call to **time(2)** and the library call to **ctime(3)**. Besides, no provision exists for correcting the value, even if it *did* change substantially between function calls.

The real reason for the passing of the address is tradition. These routines were written in the days of

COSI

C O M M U N I Q U É™



The PC-to-UNIX™ Connection

See Us At Booth #751
in Los Angeles – UNIX Systems Expo

See Us At Booth #122
in New York City – UNIX Expo

COSI

313 N. First St.
Ann Arbor, Michigan 48103
313-665-8778

Communiqué is a trademark of COSI.
UNIX is a trademark of Bell Laboratories.

```

main()          /* note how simple things were in the days of V6 */
{
    int clock[2];
    char *date;

    time(&clock);
    date = ctime(&clock);
}

```

Figure 1 — Keeping time, Version 6 style.

Version 6, before the C language had a **long** data type. Time had to be kept in a two-element array of 16-bit integers. The only way of dealing with this was to pass the address of the first array element, as in the code listed in Figure 1.

Actually, this tradition will help provide a solution in the year 2038. Time could then be held as a two-element array of integers. As a step toward this solution and general portability, many UNIX systems (starting with Version 7) keep time using the data type **time_t**, generally declared as follows:

```
typedef long time_t;
```

It would be possible to make **time_t** a structure composed of an **int** and a **long**, the first containing the year of the epoch (such as 1970), and the other containing the seconds since the beginning of the epoch. This may not be the solution adopted in 2038 (if UNIX is still around), but it is one possibility.

PUTTING THE ROUTINES TO WORK

Now, let's turn to some sample C programs. Suppose you want to print the date and time on some output, but for the sake of efficiency you don't want

to incur the extra process overhead involved in using:

```
system("date");
```

That would involve spawning a shell, which would in turn execute the **date** command. The program displayed in Figure 2 prints a simple date and time stamp, using the most basic facilities available. You could use these same lines inside a larger program.

In the example, the variable **clock** is declared as a **long** integer, which is how it's done in many programs and is the way that's recommended for System V. On systems based on Version 7, the type **time_t** may be used instead, although the **lint** library doesn't know about it yet. When **time()** is handed a null value, rather than an address (as on Version 6), it returns the time in seconds since 1970. The null value should be coerced into a pointer to a **long**, because this is what **time()** expects as its argument. Note, too, that the variable **clock** must be passed by address to **ctime()** even though its value won't be changed, because its length may vary across different versions of UNIX. The **ctime()** routine returns a pointer to a character string, which is printed out using **fputs()** rather than **printf()**, for the sake of efficiency.

```

#include <stdio.h>
#include <time.h>

main()          /* stamp - output a simple date and time stamp */
{
    long clock, time();
    char *date, *ctime();

    clock = time((long *) 0);          /* seconds time since 1970 */
    date = ctime(&clock);              /* corrected for timezone */
    fputs(date, stdout);              /* output character string */
    exit(0);
}

```

Figure 2 — Code suitable for printing date and time stamps.

UNIX™ EVALUATION NEWS!!

AIM OFFERS APPLICATION - CONFIGURABLE UNIX BENCHMARK

DO YOU HAVE TO MAKE A UNIX PURCHASE DECISION IN THE NEAR FUTURE?

If you have to compare competitive UNIX systems in the marketplace *or* even evaluate the performance of your own system in light of new enhancements available, you know it can be a bewildering task.

THERE IS MORE TO COMPARING UNIX SYSTEMS THAN VENDOR STATISTICS ALONE.

How do you decide which UNIX system should be the standard for your company? You *could* compare published vendor data sheets but you would not be sure how they support your specific needs.

UNIX SYSTEMS RANK DIFFERENTLY WHEN COMPARED IN SPECIFIC ENVIRONMENTS!!

If your target needs are heavily database oriented, obviously the disk transfer rate of each system dominates your selection criteria, but if you need graphics capability, then the CPU resource becomes the center of evaluation. You might even need to select a system which does both functions well.

A UNIX BENCHMARK TAILORABLE FOR YOUR ENVIRONMENT IS AVAILABLE.

AIM Technology's second UNIX benchmark product, SUITE II, is parameterized so that you can evaluate how various UNIX systems, (such as Version 7, System III, System V, Xenix, etc.) would perform in your processing environment. Application characteristics (such as word processing, spreadsheet, graphics, communications, compilations and scientific requirements) can be weighted to reflect how much they are a factor in your overall evaluation. This testing can be run, tuned, and re-run on as many as 20 systems for comparison...

CALL OR WRITE FOR FREE "EVALUATING UNIX COMPUTERS" MANUAL**

FOR FREE MANUAL
In California Call
800-672-3470 x 815
All Other Areas
800-538-8157 x 815

This manual discusses how to evaluate the performance of various UNIX systems in specific processing environments. It will be most valuable to the multiple-purchase buyer who must decide which UNIX system *is* best suited for the application mix representative of *his* company's requirements.



AIM TECHNOLOGY

AIM's Suite II Benchmark is available for \$3450. It can be ordered for immediate delivery from AIM Technology, 4655 Old Ironsides Drive, Suite 390, Santa Clara, CA 95050. For manual or additional information contact Ms. Jamie Mendez (408-727-3711).

™ UNIX is a trademark of AT&T Bell Laboratories

**Additional copies of the manual can be obtained for \$9.75

```

#include <stdio.h>
#include <time.h>
#include <whoami.h>
#ifdef USDL
# include <sys/types.h>
# include <sys/timeb.h>
#endif

char *day[7] = {
    "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"
};
char *month[12] = {
    "Jan", "Feb", "Mar", "Apr", "May", "Jun",
    "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"
};

main()          /* edate - date in European format (L most variable) */
{
    struct tm *t, *localtime();
    char *zone;
#ifdef USDL
    struct timeb tp;
    char *timezone();
    /*
     * On V7 and Berkeley UNIX, timezone is in timeb structure
     */
    ftime(&tp);
    t = localtime(&tp.time);
    zone = timezone((int)tp.timezone, (int)tp.dstflag);
#else
    long clock, time();
    /*
     * On AT&T's System V UNIX, timezone is in the environment
     */
    clock = time((long *) 0);
    t = localtime(&clock);
    zone = tzname[t->tm_isdst];
#endif
    printf("%2d:%02d:%02d %s %s %2d %s %d\n",
        t->tm_hour, t->tm_min, t->tm_sec, zone,
        day[t->tm_wday], t->tm_mday, month[t->tm_mon],
        t->tm_year + 1900);
    exit(0);
}

```

Figure 3 — A sample program for printing time.

LOW LEVEL LIBRARY ROUTINES

The program in Figure 2 would be adequate if you're satisfied with a simple date and time stamp in the following format:

```
Tue Jul 24 23:24:43 1984
```

But if you also want to print the time zone or want the fields to appear in a different order, you'll need to use lower level library routines. The next program we'll look at is more sophisticated, for it will actually print the name of the time zone, such as "PDT" for Pacific Daylight Time. Low-level routines to get the time are different on System V than on Version 7 and Berkeley UNIX. If you're running System III or System V, you need to place the following line in `/usr/include/whoami.h` in order for the following program to work:

```
#define USDL
```

AT&T might want us to consider its UNIX a standard, but it can't seem to decide what this standard should be called. Programmers certainly can't change from System III to System V to System V.2 every time AT&T changes the release number. Just two months ago, I urged that USG be employed to delimit code for Systems III and V. This month I'm urging that USDL be used instead. The switch owes to the recent reorganization of AT&T, during which the UNIX Support Group (USG) was renamed the UNIX System Development Laboratory (USDL). I can't predict what it will be called after the next reorganization. Maybe we should simply agree on ATnT (for American Telephone 'n Telegraph) and call it quits.

The sample program listed in Figure 3 will print the date in European format, as in:

```
23:25:08 PDT Tue 24 Jul 1984
```

In this format, fields (except time zone) vary more rapidly on the left than on the right. In order to print

the day and month as character strings, we need two arrays of character pointers, one for the days and one for the months. Both are initialized in the example as external variables. The include file `sys/types.h` is present for its definition of `time_t`, while the include file `sys/timeb.h` is present for its definition of the `timeb` structure, used by the `ftime()` system call.

On Version 7 and Berkeley UNIX, information about the time zone is kept inside the operating system. On System V, though, it's necessary to place the name of the time zone and the number of hours west of Greenwich Mean Time into the environment. This is usually done at boot time from the `/etc/rc` file, so users need not worry about it. For example, on the west coast, the `TZ` environment should be set to:

```
PST8PDT
```

The first three characters represent the name of the time zone when observing standard time, while the final three characters represent the name of the time zone during daylight savings time. The number in the middle is the hours west of GMT.

STRUCTURES

The `tm` structure is common to all varieties of UNIX, and is defined in the manner described in Figure 4 on Version 7, Berkeley UNIX and System V.

All of these values could be declared `short` instead of `int`, but traditionally they are stored as integers. In the program listed in Figure 3, the `localtime()` library function returns a pointer to one of these `tm` structures.

The `timeb` structure is used on Version 7 and on Berkeley UNIX, but not on System V. It is smaller than the `tm` structure, but contains a variety of types, as shown in Figure 5. The reason for using the typedef `time_t` was explained above. More precise

```
struct tm {
    int      tm_sec;      /* seconds (0-59) */
    int      tm_min;      /* minutes (0-59) */
    int      tm_hour;     /* hours (0-23) */
    int      tm_mday;     /* day of the month (1-31) */
    int      tm_mon;      /* month of the year (0-11) */
    int      tm_year;     /* current year - 1900 */
    int      tm_wday;     /* day of the week (0-6) */
    int      tm_yday;     /* day of the year (0-365) */
    int      tm_isdst;    /* 1 if daylight savings time */
};
```

Figure 4 — Definition of the `tm` structure.

```

struct timeb {
    time_t  time;           /* seconds since 1970 */
    unsigned short millitm; /* a more precise interval */
    short   timezone;      /* minutes westward from GMT */
    short   dstflag;       /* 1 if daylight savings time */
};

```

Figure 5 — The varied types of *timeb*.

intervals than seconds are generally measured in milliseconds. The reason the time zone is represented in minutes, rather than in hours, is that Afghanistan time is 4 1/2 hours east of GMT. On System V, this might represent a problem, because the **TZ** environment can't handle portions of an hour.

System V time zone names are held in an external, two-dimensional array. The first is the name of the time zone when observing standard time, while the second is its name during daylight savings time. In the program listed in Figure 3, the field **tm_isdst**

in the **tm** structure is used as an index into this two-dimensional array.

Finally, the date and time are output with a giant **printf()** statement. This program will continue to work after the year 2000, because we add 1900 to the value **t->tm_year**, rather than embedding the string "19" into the print format.

THE BOTTOM LINE

There are two quick lessons on C programming to be learned here. First of all, parameters should be passed by address when the manual page says a function is expecting a pointer. When the programmer's manual documents a function call like:

```

ftime(tp)
struct timeb *tp;

```

it means that your program has to pass the address of that structure to the function, as in the line:

```

ftime(&tp);

```

That's because the address of a variable is in fact a pointer to the variable.

Second, when a function call returns a pointer, you don't need to allocate space for the thing that's pointed to in your program. In the example in Figure 3, we allocated space for the **timeb** structure, but allocated only a pointer for the **tm** structure. This is because **ctime()** and **localtime()** return pointers to a **tm** structure, whereas **ftime()** returns nothing. We have to allocate the entire **timeb** structure and pass its address to **ftime()**, which will fill in the appropriate values. If you don't need to allocate space, it's best to use a pointer because that will make your program both more efficient and more compact.

Bill Tuthill was a leading UNIX and C consultant at UC Berkeley for four years prior to becoming a systems software analyst at Imagen Corporation. He enjoys a solid reputation in the UNIX community earned as part of the Berkeley team that enhanced Version 7 (BSD 4.0, 4.1 and 4.2). ■

ACUITY® business software is compatible with any budget, and all these systems:

UNIX-based Micros	VAX, VMS or UNIX
PRIME	Convergent
IBM-PC	Harris

With prices from \$700 to \$6500 for a fully supported package, any size company can afford our general accounting and specialized project cost software.

Packages available include project management, labor/ODC forecasting, work breakdown structure, customer order processing, bill of materials processing and inventory management.

Plus a complete set of accounting software including general ledger, payables, receivables, payroll and fixed assets.

Call (619) 474-2010 for details.

See us at UNIX SYSTEMS EXPO/84, Los Angeles



225 West 30th Street, National City, California 92050

Circle No. 12 on Inquiry Card



KEEPING YOUR SYSTEM HEALTHY

A prescription for keeping your
system fit

by Jim Joyce and Douglas R. Merritt

Most of the lore about UNIX system administration passes from person to person as part of the general oral tradition for learning about the system. This article commits a collection of techniques to writing and demonstrates their usefulness with examples captured with Mark Horton's **script** utility. Readers are encouraged to put the examples online and make them part of their system administration toolkits.

CHOOSING AMONG SHELLS

There are two major shells, or command interpreters, available under UNIX. One is the standard UNIX shell, **/bin/sh**, often referred to as the Bourne shell for its author, Dr. Stephen R. Bourne. The other is the UC Berkeley C shell, **/bin/csh**, written by Bill Joy and enhanced by a number of others. Each shell excels at different tasks.

The greatest advantage the Bourne shell enjoys over the C shell lies in its brevity and speed. Comparable scripts run from four to six times faster under the Bourne shell on computer systems that do not use virtual memory for paging. Thus the standard shell is better for developing system administration shell scripts, such as the one in **/etc/rc**. Also, it is more portable than the C shell. Every UNIX system since Version 7 has the Bourne shell, though only systems with Berkeley enhancements have the C shell. Certain system commands, such as **calendar** (to be discussed below), are in fact Bourne shell scripts.

The C shell, on the other hand, provides a better environment for program development and interactive system administration. Its **history** mechanism allows users to recall commands run during the current session. So if a problem surfaces



that might have been caused by a previously issued command, the built-in shell command:

```
# history
```

will list previously-issued commands for inspection. This feature alone makes it worthwhile to invoke the C shell, even when it is not the login shell for root.

Some manufacturers make the C shell the login shell for the root account, with a **history** setting of 20. But for intense system maintenance work, such as moving user accounts or tuning the system for performance, a better setting would be:

```
# set history = 100
```

The C shell will then remember the last 100 commands, a much more complete list.

It is best to perform system administrative tasks in single-user mode. Single-user mode allows the administrator to operate as root under the Bourne shell. To switch to the C shell, enter:

```
# csh
# source /.csh
```

to put the C shell's tailoring to work for the root account.

FILES THAT GROW

There are five files in particular that can grow unnoticed to fill a disk. Their purpose is to help monitor activity on the system at points where security might be breached. They need to be trimmed so they do not become more cumbersome than useful. The files are listed in Figure 1.

All users can access the file `/usr/adm/wtmp` by

using the **last** command. For example, to find when *jim* was last logged in, enter:

```
% last jim
```

In our case, the result was:

```
jim ttyd5 Mon Jul 9 09:10 still logged in
jim ttyd5 Sun Jul 8 14:20 - 18:11 (03:51)
jim console Sun Jul 8 13:34 - 13:35 (00:01)
```

If an account that has presumably been inactive shows recent activity, you may have had an intruder. Another place to look for evidence of infiltration is in the system's log of bad logins. Use of the **who** command on the files in `/usr/adm/date` requires that the user either be in the `/usr/adm/data` directory or employ a full pathname. Hence, an outsider will likely attempt a number of bad login names before hitting on a correct one. To find all bad login names, enter:

```
% who /usr/adm/data/badname
```

or:

```
% cd /usr/adm/data; who badname
```

To see valid accounts for which bad passwords were entered:

```
% who badpw
```

or:

```
% who /usr/adm/data/badpw
```

Checking the contents of these files periodically may reveal attempts to breach system security.

Some or all of the files shown in Figure 1 may be missing, and if so, they must be created manually since the programs that write accounting data to them only *add* to existing files.

File	Purpose
<code>/usr/spool/uucp/LOGFILE</code>	Logs actions by the uucp system
<code>/usr/adm/wtmp</code>	Records all logins since the file's creation
<code>/usr/adm/data/badname</code>	Gives incorrect login names
<code>/usr/adm/data/badpw</code>	Stores incorrect passwords entered for valid login names
<code>/usr/adm/data/tracct</code>	Logs troff usage

Figure 1 — Five files that may need trimming.

The file `/usr/spool/uucp/LOGFILE` should be owned by `uucpadmin` and belong to the `uucpadmin` group as well. The other files should be owned by `bin`, and should belong to the `bin` group. Clearly, each of the five files need to be write-protected from other users.

Growing files can be trimmed as a matter of system maintenance by copying `/dev/null` to them just *after* an epoch-level dump of the file system they're stored on. This way, the data is saved on the backup media before the files are trimmed — just in case they are needed later.

SECURITY ISSUES

No one likes to think about security breaches, but if you have so much as one telephone line offering a login to outsiders, you have a potential security breach, no matter how dependable and trusted the known users may be. Unwitting or unfriendly strangers may dial in by accident, and when they do, they should not have access to sensitive files.

There should be no user login without a password. Precaution dictates that passwords not be words that can be found in a dictionary. Neither should they be nicknames, initials, spouse names, vehicle license plate identifications or or any other easily discovered word. A minimum of six characters should be used, including at least one non-alphabetic character.

POINTERS ABOUT `fsck`

The `fsck` program is an invaluable tool for system administrators, though its proper use is less than clear from reading standard documentation. It is reasonably smart and, under thoughtful guidance, will clear up many file system maladies. Here we present some additional benefits.

Because a file system becomes more fragmented through the addition of new files and the deletion of old ones, system performance will slow down over time, a phenomenon especially noticeable to `vi` users. The best solution to file fragmentation is to perform a full **dump** and **restor** of the file system. But if the backup media is not totally reliable, there is the risk that some data may be lost. Thus it is not a good idea as a matter of course to tune up a file system with a full **dump** and **restor**.

However, there is an option to the `fsck` command, the `-S` option, that will conditionally reorganize the free list, as shown in Figure 2. The manual page for `fsck` shows a lower case `-s` option, which unconditionally reorganizes the free list, but a cautious ap-

```
# fsck -S /dev/rgb0c

/dev/rgb0c
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Cneck Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Free List (Ignored)
** Phase 6 - Salvage Free List
2330 files 13089 blocks 4328 free

***** FILE SYSTEM WAS MODIFIED *****
#
```

Figure 2 — *The workings of `fsck -S`.*

proach to such activities is best. Upper case **S** allows for user interaction and so serves as that cautious approach.

Reorganizing the free list is a valuable and safe activity. Through the normal system activity of creating and deleting files, the free list becomes fragmented. New files created using the fragmented list stand a good chance of being fragmented themselves. Using the `-S` option for `fsck` is the best step short of doing a **restor** for allowing the system to make the best use of available space. Programs that use disk space for data buffering, such as `vi` respond more quickly after the free list has been reorganized.

Among the options given on the manual page for `fsck` is the `-y` option. This option should *never* be used! Although `fsck` is an intelligent program and usually can be entrusted with fixing file system irregularities, there are circumstances in which it will become confused and suggest removing certain files it should not remove, such as `/unix`. The mere potential of this occurring is reason enough to shun the `-y` option.

Many UNIX systems can be left up round-the-clock for indefinite periods of time. The system on which this article was written has the record listed in Figure 3 for its last six reboots. This information was gathered by the command:

```
% last reboot | head -6
```

For systems that are up for extended periods it is wise preventive maintenance to run `fsck` weekly to check the integrity of the file system. Despite its soundness as an operating system, UNIX does have lurking problems that affect the file system in unex-



```

reboot      ~           Sun Jul  1 12:46  still logged in
reboot      ~           Wed Jun 27 19:12  - crash (3+17:34)
reboot      ~           Wed Jun 27 18:48  - crash (00:23)
reboot      ~           Mon Jun 18 18:17  - crash (9+00:31)
reboot      ~           Sat Jun 16 15:33  - crash (2+02:44)
reboot      ~           Sat Jun 16 15:20  - crash (00:12)

```

Figure 3 — A sample log of system startups.

pected and unwelcome ways. Regular use of **fsck** for file system maintenance is much like flossing for dental hygiene.

USING CALENDAR

The **calendar** utility is an electronic reminder service for important dates and appointments. As such, it can be a helpful tool to a system administrator. Although **calendar** does know about weekends, it does not know about holidays or events. However, date-stamped messages about holidays and scheduled activities affecting users can be stored in a file named **calendar** in the administrator's login directory where the utility will find it.

An entry in **/usr/lib/crontab** activates the calendar program once a day to mail users the messages they have stored in their own calendar files. A sample **crontab** entry is:

```
0 4 * * * /bin/calendar -x
```

which means that the program will be run each day at 4 am.

The manual page for **calendar** incorrectly states that the date can be noted in several forms. On many systems, the only form the **calendar** program recognizes is that of month and day, such as "Jul 4" for "July 4." A sample entry is:

```
Jun 28 Alert users to 4th of July holiday hours
Jun 29 Post one week notice of new dialup number
```

The **calendar** shell script was written by Doug Mellroy at Bell Laboratories, and is worth study for the shell programming techniques it uses.

USING FIND

The hierarchical file system, one of the most powerful UNIX features, is also the root of much evil for users and administrators alike. Anyone acquainted with the system knows how trying it can be to locate a specific file in the midst of hundreds of subdirectories.

The **find** command searches subdirectories for files with specified characteristics and can be used to search the entire file system. For instance, the following command finds a file named **lstree.c** where the full pathname is not known:

```
% find / -name lstree.c -print
/a/usr/doug/cutil/lstree.c
%
```

Figure 4 explains each of the arguments to this command. Note that the **-print** flag is necessary to get any results printed. The default action for **find** is to do nothing with the files it finds, which is not usually what is wanted. Further, it says nothing at all when it cannot match a specification. This lack of communication can be confusing. All the same, **find** is a useful command.

Argument	Explanation
/	directory in which to start search
-name	look for file with following name
lstree.c	name to look for
-print	print names of any files that match specification

Figure 4 — Analysis of command for finding a file whose pathname is unknown.

A NEW DIMENSION IN DATABASE MANAGEMENT



Booth #1996-7

See us at

COMDEX™/Fall '84

November 14-18, 1984

Las Vegas Convention Center
Las Vegas, Nevada

See us at UNIX SYSTEMS EXPO/84, Los Angeles,
September 11-14, Booth #928

MISTRESS is the fully relational database management system (RDBMS) for UNIX.* It features the Structured Query Language (SQL*) for the end user as well as standard programming interfaces to the C language for the DP professional. Advanced concepts include variable-length character fields, dynamic storage allocation, and B+ Tree indexing. **MISTRESS** has been designed exclusively for the UNIX environment and is totally written in C.

MISTRESS/32 is the advanced relational database management system for extended addressing UNIX products. **MISTRESS/32** features enhanced capabilities for security, recovery and data integrity, as well as a fully integrated report writer and screen interface. **MISTRESS/32** is the recommended system for more demanding applications.

*UNIX is a trademark of Bell Labs. IBM and SQL are trademarks of International Business Machines.

RHODNIUS Incorporated

10 St. Mary Street, Toronto, Ontario, Canada M4Y 1P9

(416) 922-1743

Telex: 06-986766 TOR.



Argument	Explanation
/	directory in which to start search
-name	look for file with following name
core	name to look for
-atime	file must also have an access time as specified by the following argument
+7	access time must be greater than or equal to 7 days
-print	print pathnames for any files that match specification

Figure 5 — Analysis of a command for finding dated files.

Argument	Explanation
/	directory in which to start search
-perm	look for file with following permissions
-4000	set user id bit as specified in chown(2)
-a	conjunction of two specifications; no longer necessary but still often seen
-user	file must also be owned by following user
root	owner of file
-print	print results of search

Figure 6 — Sleuthing for possible security breaches.

An example follows for searching for files with a given name that may match several files:

```
% find / -name core -print
/a/usr/doug/cmisc/core
/a/usr/doug/yacc/core
/a/usr/mark/review/core
/a/usr/dave/ubs.forms/core
%
```

This can be useful for finding files to delete when a system is critically low on disk space. In some instances a system may have dozens or even hundreds of old **core** files, which can waste megabytes of disk space.

More often than not, **core** files that are more than a week old should be considered as prime candidates for removal, like so:

```
% find / -name core -atime +7 -print
/a/usr/doug/yacc/core
/a/usr/mark/review/core
%
```

An explanation of this command can be found in Figure 5.

Another use of **find** is to look for all files on the system that are "suspicious," such as files for which the **set userid** permission allows the user to become **root** — a security hold that could give unauthorized people superuser privileges. There are only a few UNIX utilities where the **set userid** permission will allow one to become **root**. Unfamiliar files showing this permission setting may indicate illicit activity on the system. See the box below for an example. An explanation of the command components can be found in Figure 6.

The system administrator on the system from which this example was taken might want to discuss security issues with user **doug**.

```
% find / -perm -4000 -a -user root -print
/a/usr/doug/secret/shell
/bin/at
/bin/mail
/bin/mkdir
/bin/mv
/bin/newgrp
/bin/passwd
/bin/rmdir
/bin/su
/bin/rmail
/bin/lpr.std
/usr/bin/cu
/usr/bin/mf0
/usr/bin/uf0
%
```

Using **find** to locate security holes.

FILES THAT ARE OLD

The command shown in Figure 7 can be used to search for little-used files that are candidates for archiving. It will find and do a long listing of all files in *jim*'s account that have not been accessed in over 90 days.

It may be desirable to ask users regularly to delete or archive old, seldom-accessed files. The Bourne shell script in Figure 8 will find all user files on the system that have not been accessed in 90 days, sending a mail message to their owners listing the filenames considered as candidates for archiving.

The list of users in the script is obtained from **/etc/passwd** using **sed**. Standard system accounts are ignored using the shell's **case** command. Each user's home directory is tested to make sure it exists before **find** is used to locate old files. The output from **find** is put in a temporary file. If this temporary file is not empty — that is, if **find** succeeded in finding files not accessed in 90 days — the list of old files is mailed to the owner. A message is then sent explaining the list of files. The mail is sent this way because the **mail** program displays the most recent mail message first.

FINDING ZERO LENGTH FILES

Zero length files actually do take up space in a file system. The directory entry consumes space, as do possible links and space for inode entries. Although such amounts of space are not large, they are clearly wasted. If the available inode entries are exhausted for a given file system, no new files can be created even if space for the data is available. To find zero length files, use a command like the following example:

```
% find ~jim -size 0 -print
/a/usr/jim/weekend/courses/titles
/a/usr/jim/zero
```

An explanation of the command can be found in Figure 9. Zero length files are safe to remove, and **find** can be used to do the job:

```
% find ~jim -size 0 -exec /bin/rm "{}" ";"
```

An explanation of the options can be found in Figure 10.

There is reason for using the two-step approach employed here for finding and removing zero length files. Various utilities, such as **lpr** and **uucp** use zero length files as a means of locking a device so that two or more utilities do not attempt to send output to the

Introducing TI's Climb on the 32-



Nu Machine. bit NuBus now.

The Nu Machine™ Computer. The first system in the Texas Instruments Nu Generation Computer family. The only system now available built on a modern 32-bit bus. The processor-independent NuBus™ architecture helps meet your advanced-technology design requirements today. And tomorrow.

First high-performance 32-bit bus

The NuBus technology, designed at M.I.T., is optimized for 32-bit data and address transfers. Its 37.5-Mbyte/sec bandwidth combines with an elegant arbitration scheme to ensure fast and fair data flow.

Innovative, flexible architecture

The NuBus design was developed to support sophisticated system architectures and eliminates the

built-in obsolescence of processor-dependent systems. It lets you concentrate on developing applications, not architecture. Your significant investments are protected as new technologies develop.

The Nu Machine's open architecture solves your make vs. buy dilemma. Multiple-processor configuration support combines with the NuBus high bandwidth, high-resolution graphic displays, cache memory, and high-speed disks to make the Nu Machine system attractive to sophisticated end-users, systems integrators, and OEMs in the engineering and scientific marketplace.

Anticipating industry trends, the power and expandability of TI's Nu Machine allow it to accept 32-bit processors of the future.

Open system supporting industry standards

TI's Nu Machine system is currently available with a

10-MHz 68010 processor supporting a UNIX™-based operating system with enhancements for windowing and high-resolution displays.

Those who want to design their own system processors and controllers can now license the NuBus design from Texas Instruments.

Also, a NuBus-to-Multibus™ converter allows the use of existing interface cards and peripherals from third parties.

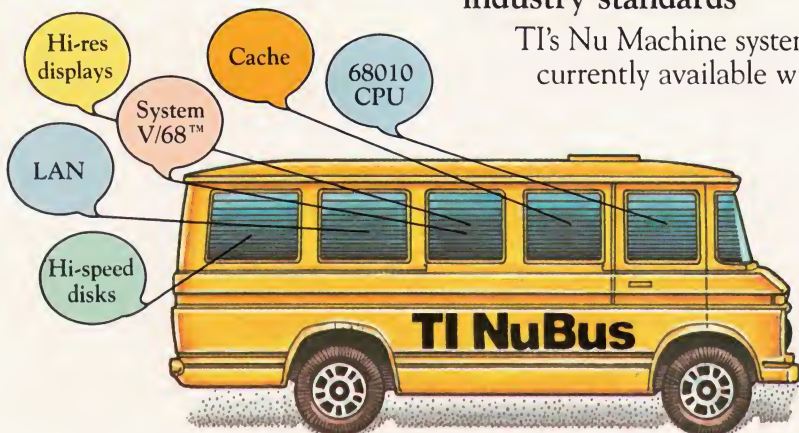
The system you can build on from now on

Because its high performance and flexibility are designed for the long run, TI's Nu Machine can be updated when other systems are outdated.

And, Nu Machine computers are backed by TI's service and customer-support network and by TI's commitment to quality and reliability.

To climb on the NuBus bandwagon, call toll-free: 1-800-527-3500. Or write Texas Instruments Incorporated, P.O. Box 402430, Dept. DNA203UR Dallas, Texas 75240.

Nu Machine and NuBus are trademarks of Texas Instruments Incorporated
Multibus is a trademark of Intel Corporation
System V/68 is a trademark of Motorola, Inc.
UNIX is a trademark of Bell Laboratories



Combining innovative NuBus architecture with advanced graphics, powerful peripherals, and UNIX-based software, TI's Nu Machine provides the outstanding performance and flexibility required by scientific and engineering systems designers.

**TEXAS
INSTRUMENTS**
Creating useful products
and services for you.



```

% find ~jim -atime +90 -exec ls -l "{}" ";"
-rw-r--r-- 1 jim      129 Mar 13 19:51 ./login.ok
-rw-r--r-- 1 jim      754 Dec 23  1983 ./lasttest
-rw-r--r-- 1 jim      895 Feb  2 17:34 ./tab300s.c
-rw-r--r-- 1 jim      374 Apr 15 15:24 ./mail/bobt
%

```

Argument	Explanation
~jim	directory in which to start search (jim's home directory)
-atime	look for file with following access time
+90	last access time must be greater than or equal to 90 days ago
-exec	execute the following shell command on each file that matches the preceding specification
ls	name of command to execute
-l	argument to command (for long listing format)
"{}"	name of file which met the find specification; command will be run once for each matching file
";"	a quoted semicolon ends the command specified by -exec

Figure 7 — Searching for archive candidates.

Argument	Explanation
~jim	The directory in which to begin the search
-size	look for file with the following size
0	empty files, which are zero length
-print	print results of search

Figure 9 — Analysis of a command for finding zero length files.

device at the same time. These lock files should not be removed if a line printer is in use or **uucp** is active.

System administration can be enhanced by a collection of techniques such as the ones described here. Our descriptions are meant to be thought-provoking — not exhaustive.

This article is an adaptation of the chapter, "Making System Administration More Productive," from International Technical Seminars' System Administration Manual. Copyright 1984 by International Technical Seminars, Inc., 520 Waller Street, San Francisco, CA 94117, 415/621-6415.

```

:
# moldy.oldies -- an administrative script to be run weekly
# to search each user's file space for files that have not been
# accessed in over 90 days.
# (c) 1984 by International Technical Seminars, Inc.
# 520 Waller St., San Francisco, CA 94117 (415) 621-6415.
# Permission to copy is granted if this copyright notice is
# included in its entirety. Adapted from the /bin/calendar script
# by Doug McIlroy of Bell Laboratories.

PATH=/bin:/usr/bin
sed '
    s/:]*):.*):[^:]*$/y=2 z=1/
    read x
do
    eval $x
    trap "/bin/rm /tmp/$z.$$; trap '' 0; exit" 0 1 2 3 13 15
    case $z in
    adm)          ;;
    bin)          ;;
    check)       ;;
    daemon)      ;;
    root)        ;;
    root2)       ;;
    rootsh)      ;;
    sys)         ;;
    uucp)        ;;
    uucpadm)     ;;
    who)         ;;
    *)
        if (test -s $y) then
            cd $y
            find . -atime +90 -print 2>/dev/null > /tmp/$z.$$
            if (test -s /tmp/$z.$$) then
                mail $z < /tmp/$z.$$
                mail $z <<++
                Dear $z :
                The next mail message you see will contain
                names of moldy oldies you have not accessed
                in 90 days. Please examine these files to see
                if they can be archived.
                -- Your Friendly System Administrator
            fi
            /bin/rm /tmp/$z.$$
        fi
    esac
done

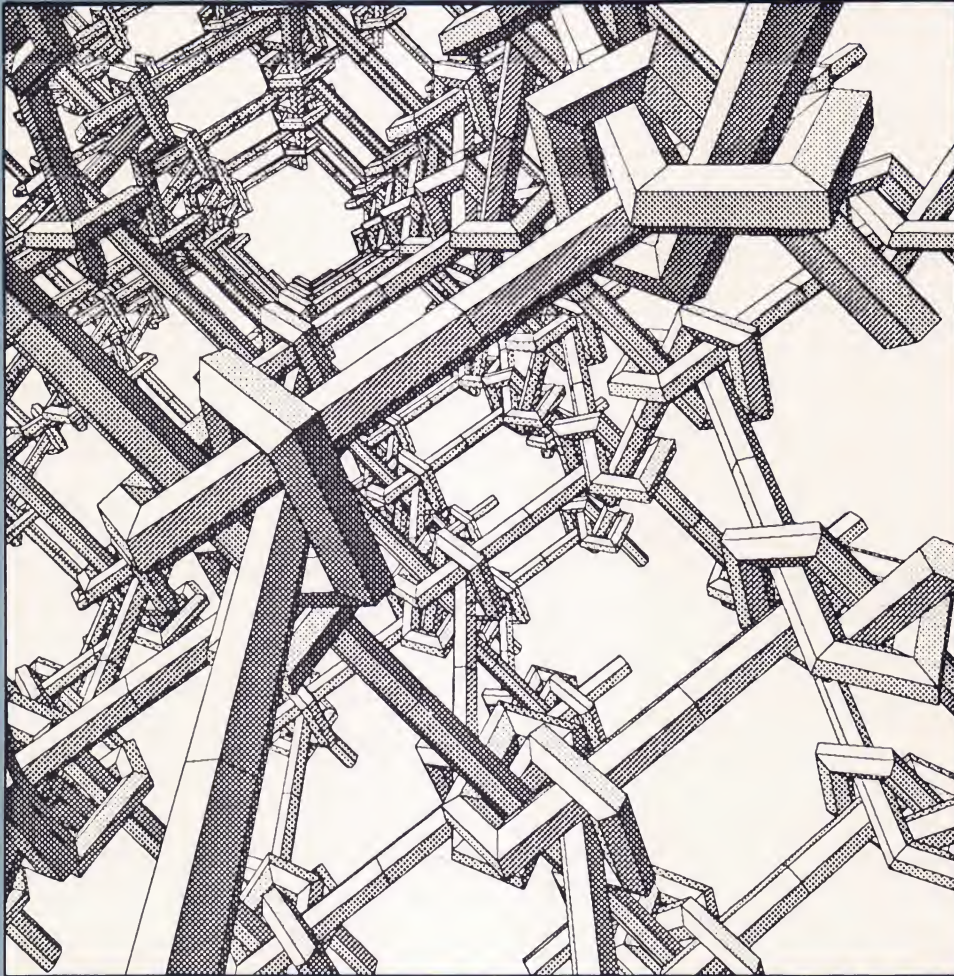
```

Figure 8 — A shell script for finding old “moldy” files.

Argument	Explanation
-exec	execute the following command if a file is found that is zero length
/bin/rm	remove the file
"{}"	quoted braces that fdind replaces with the pathname of the file it found
";"	quoted semicolon required for the shell command

Figure 10 — Analysis of a command for removing zero length files.

LINKING UP



WITH THE OUTSIDE WORLD

An introduction to UUCP

by Bob Toxen

Officially, **UUCP** is short for **UNIX to UNIX** copy program. But in addition to a program for copying files between UNIX computers, **UUCP** offers users a means of executing programs remotely and sending network mail. Because of this, I prefer to think of **UUCP** as meaning **UNIX to UNIX** communications package.

The program to copy files is called **uucp** and works similarly to the **cp** utility. It takes two (or more) arguments and copies the first file (and any subsequent files) to the last filename (which may be a directory). Each argument, except the last, should be the name of a file and may be preceded by a system name. The system name should be separated from the filename by an exclamation mark (!), pronounced by UNIX buffers as "bang."

For instance, to copy the file *flight* from your local system to one called *xorn*, give the command:

```
% uucp -m -njill flight xorn!\~jill/flight
or:
```

```
$ uucp -m -njill flight xorn!\~jill/flight
```

The first command line would be used with the C shell while the second would be used with the Bourne shell. As you might have guessed, I threw in a few tricks. First, the **-m** flag tells **uucp** to send you mail when the copy finishes. This is important because **uucp** merely queues up a request to do the copy. The copy operation itself may not occur for several hours depending on how **uucp** is configured (on both systems) and what phase the moon is in.

The second trick I used, **-njill**, will cause **uucp** to send mail to the account called *jill* on the other system (named *xorn*) when the operation is done. Third, since I do not know Jill's home directory path, I specified **~jill**. This will be converted to her home directory's full pathname on *xorn* by **uucp** and so will work with both the C shell and the Bourne shell. C shell users must precede the ! separating the system name from the account name with a backslash (\) since C shell normally treats exclamation marks as special characters.

SECURITY AND PERMISSIONS

A common problem with using the **uucp** command appears in the area of permissions. Not only must users tussle with the usual assortment of UNIX file system permissions — they must also thread their way through an additional group of requirements. The source file *flight* in our example,

must be readable by everyone. Also, the directory it is in (our current directory in this case) and all directories leading to it must be readable and executable by everyone. The destination file — if it exists — must also be writable by everyone. If a destination file does not exist, the directory in which you intend to create a new file must be writable by everyone. In any case, all directories leading to the file you create or modify must be readable and executable — just as the source directories must be.

Most people do not want their login directories writable by everyone because if they were, anyone could remove files, either by accident or on purpose. I solve this problem by creating a subdirectory under my login directory called **uucp**, which I make writable by all. Also, I make sure that any confidential files are not readable by others so as to prevent someone on another system from using the **uucp** command to copy confidential material to their system.

Because many users are not computer security experts and do not want to worry about file permissions, many system administrators configure **uucp** to allow only file transfers to and from pathnames beginning with **/usr/spool**. The directory **/usr/spool/uucppublic**, which is readable, writable and executable by all, is provided for users to send and receive files. Thus to send the file *flight*, one would give the commands:

```
% chmod 755 flight
% cp flight /usr/spool/uucppublic
% uucp -m -njill ~uucp/flight xorn!\~uucp
```

In this case, we assume the C shell is used. There should be an account called **uucp** on each system with a login directory of **/usr/spool/uucppublic** so that the C shell will know how to expand **~uucp** on the local system. Any occurrences not expanded by the shell will be expanded by **uucp**. Jill would then enter:

```
% mv ~uucp/flight .
```

to move the file into her login directory.

REMOTE PROGRAM EXECUTION

The program called **uux** is used for remote program execution. You can invoke one program or you can invoke several by having the output of one program piped to the input of the next. Standard input and standard output may be redirected to and from files on other systems. For example, if you are on the



system called *sauron* and you want to know who is logged into the system called *xorn*, you could issue the command:

```
% uux "xorn\!who > sauron\!~uucp/who.xorn"
```

The program **uux** will cause the **who** command to be invoked on *xorn* and will accumulate the output in a temporary file before using **uucp** to transfer that file back to the file `/usr/spool/uucppublic/who.xorn` on *sauron*. Alternatively, you could mail the results back, like so:

```
% uux "xorn\!who | rmail (sauron\!bob)"
```

Note the parentheses around `sauron\!bob` to tell **uux** not to interpret it as an *input* file. Otherwise, **uucp** would see `sauron\!bob` as an argument to **rmail**. It would be instructive to issue the commands listed in Figure 1 and examine the results (even if your system is not configured to allow them).

We can do more sophisticated operations — such as printing files from many systems — like so:

```
% uux "xorn\!lpr sauron\!~bob/fo  
dragon\!/usr/007/sphere"
```

Most administrators only allow certain commands to be executed via **uux** for security reasons.

Sending remote mail is very easy. To send mail to a remote system, simply give the system name and the account name separated by an exclamation mark. Thus, to send mail to *jim* on system *its*, give the command:

```
% mail its\!jim  
What did you think of that talk? Pretty good, huh?  
I'll be waiting to hear from you.
```

```
Bob  
^D  
%
```

Unlike the other commands, **mail** can send data through intermediate systems. Thus, I could give the command:

```
% mail olympus\!ucbvax\!dual\!fair  
When will the next release of the  
net-news software package be ready?  
Will you be at UniOps?
```

```
Bob  
^D  
%
```

This will send the message to *olympus* which will forward it to *ucbvax*, which in turn will forward it to *dual*, where it finally will be placed in the *fair* account.

Under System V, the restrictions against using **uucp** to transfer files to and from systems other than

those that talk directly to yours have been lifted. The file must originate in `/usr/spool/uucppublic` and be sent to the same directory on the destination system. These indirect transfers may be controlled with the **ORIGFILE** and **FWDFILE** files in `/usr/lib/uucp` so as to keep bad guys from getting into your system.

NET ADDRESSES

To send mail to someone, all you need to know is the path to that person's system. Most people know the path from their system to one of the major mail exchanges, called *backbone* sites. The paths between these installations are well known so you should be able to derive a path to the system you are trying to send mail to if you can simply learn the path from it to one of the backbone sites.

For example, my path from *ucbvax* is: `ucbvax!Shasta!olympus!bob`. If I want to send mail to someone whose path from *ihnp4* is `ihnp4!mitse!bonzo!ronnie`, I would give the command:

```
% mail Shasta\!ucbvax\!ihnp4\!mitse\!bonzo\!ronnie  
<text of message >  
^D  
%
```

The UUCP network is connected to various other networks. For example, *olympus* is the gateway to the Silicon Graphics Ethernet, which has dozens of workstations, smart graphics terminals and VAXen communicating via IP/TCP and XNS at 10 Mbaud. The UUCP network is also tied into the ARPANET at numerous sites.

Addresses are specified differently on ARPANET. Or, each ARPANET system, the paths to every other site are stored (there are far fewer ARPANET outposts than UUCP sites, which numbered at least 3000 at last count). ARPANET users also use an *at-sign* (`@`) instead of a *bang* character (`!`) and put the user name to the left of the system name.

Thus, an account called *berch* on a system called *LLL-TIS* would be referenced as `berch@LLL-TIS`.

```
%echo hello > ~uucp/fo  
% uux "xorn\!echo sauron\!~uucp/fo > sauron\!~uucp/one"  
% uux "xorn\!echo (sauron\!~uucp/fo) > sauron\!~uucp/two"  
% uux "xorn\!ls -l sauron\!~uucp/fo > sauron\!~uucp/three"  
% uux "xorn\!ls -l (sauron\!~uucp/fo) > sauron\!~uucp/four"  
%/usr/lib/uucp/uucico -r1 -sxorn
```

Figure 1 — Illustrative UUCP commands you may wish to try yourself.

If I wanted to send mail to *berch*, I would have to send the letter to a *gateway* system that understands the ARPANET syntax. One path might be:

```
% mail Shasta\!berch\@LLL-TIS
```

Here, we use *Shasta* as the gateway. Since our system, *olympus*, does not understand the ARPANET syntax, it just sends the whole mess to *Shasta* for sorting when it sees the ! after that site name.

Note that a site does not actually have to be on the ARPANET to know how to deal with it. We could even teach *olympus* to recognize *at-signs* and know that it talks to a station on the ARPANET (*Shasta*). We could then give the command:

```
% mail berch@LLL-TIS
```

and the *olympus* mailer could convert it to:

```
% mail Shasta\!berch@LLL-TIS
```

But *olympus* runs 4.2 BSD and does have the software to do this.

A *domain* name is sometimes appended to the

address. A domain, essentially, is the name of a network containing a number of systems. In mail's "From" line, it is preceded by a period (.) and is usually listed in capital letters. Thus a letter might be from:

```
olympus!bob@su-shasta.ARPA
```

If you wanted to send mail to this person from your system you would throw away the *.ARPA*, knowing that *su-Shasta* is the ARPANET name for a gateway to the UUCP network whose UUCP name is *Shasta*. Thus it translates to:

```
Shasta!olympus!bob
```

Note that the first form (without *.ARPA*) would be used to send mail from an ARPANET site.

Bob Toxen is a member of the technical staff at Silicon Graphics, Inc. He has gained a reputation as a leading uucp expert and is responsible for ports of System V for the Zilog 8000 and System III for the Motorola 68000. ■

Your PC can now "speak" C when using...

BASTOC™

A BASIC-To-C Translator

JMI's BASTOC is a versatile software tool which converts BASIC source programs to C source programs. BASTOC also translates multiple dialects of BASIC.

Registered Trademarks: IBM—International Business Machines Corp., Microsoft, MS, XENIX—Microsoft Corp.; Applesoft—Apple Computer, Inc.; TRS-80—Tandy Radio Shack Corp.; CBASIC—Digital Research, Inc.
Distributors: **Australia**, Fawhray Pty. Ltd., P.O.B. 224, Hurstville, NSW 2220 (612) 570-6100; **Japan**, Advanced Data Controls, Corp., Chiyoda-ku, Tokyo (03) 263-0383; **United Kingdom**, Real Time Systems, Newcastle upon Tyne, 0632-733131

JMI SOFTWARE CONSULTANTS, INC.
P.O. BOX 481 • 904 SHEBLE LANE
SPRING HOUSE, PA 19477 • 215-628-0846

BASTOC FEATURES

- Supports Microsoft BASIC or DRI CBASIC.
- Full run-time support library is provided.
- Provides conventional BASIC compiler when used in conjunction with a C compiler.
- Produces formatted and structured C code, easily maintained and modified.
- The output of BASTOC may be used directly as input to a standard C compiler.
- Several dialects of BASIC may be translated to C on the same system.
- BASTOC will soon translate additional dialects, including SMC BASIC.

AVAILABLE NOW

JMI's BASTOC is available now for the IBM PC and PC-compatible computers using PC-DOS or MS-DOS, and the Radio Shack TRS-80, Model 16, running XENIX. BASTOC will be available for other systems soon!

ORDER NOW

The BASTOC single unit binary price is \$350, and includes documentation, media, and shipping. To order, or to get more information, write or call JMI Software Consultants, Inc., 215-628-0846. Check, Money Orders, VISA and MC are acceptable.

◆ TERM CAP UNVEILED ◆

Excerpts from a wizard's handbook

by Douglas R. Merritt



When Bill Joy first wrote the **vi** editor at UC Berkeley, he hard-wired it to support only Lear Siegler ADM 3A terminals since they were the ones most commonly used on campus at the time. Within a very short period of time, though, Joy was deluged with requests for **vi** support for many other terminals, and so **termcap** was born.

Short for **terminal capabilities**, **termcap** is a database describing the functions commonly performed by terminals. It allows **vi** (or **rogue**) to support a new type of terminal with only the addition of a few lines of cryptic description to the file `/etc/termcap`.

Initially available only in Berkeley releases of UNIX, **termcap** quickly spread to most UNIX systems. Yet another terminal capability database named **terminfo** is on the horizon. Designed by Mark Horton, who maintained **vi** and **termcap** for a time at UC Berkeley, it is similar in many respects to **termcap**. But while **terminfo** may eventually supplant **termcap**, it is not yet in wide circulation, and so will not be addressed in this article.

Each terminal function described in **termcap** appears as a somewhat mnemonic two-letter capability name followed by a sequence of characters the terminal recognizes as the description of a particular function. Each capability field for any given terminal is separated from other fields by colons. Most terminal descriptions have as many fields crammed onto a single line as can possibly fit. The net effect is to cause **termcap** descriptions to resemble strongly

the sequences of garbage characters sometimes seen on dialup terminals after phone lines have been hit by lightning. Though the entries do become more readable after a certain



amount of practice, the problem of poor documentation remains. Most of it is still in first draft form and contains numerous omissions and errors. Writing and maintaining **termcap** definitions thus remains a black art.

SCREEN EDIT FUNCTIONS

Typical screen edit functions include *insert line*, *delete line*, *insert character* and *delete character*. Terminals that perform these functions are usually called "intelligent," even if their behavior is otherwise moronic. Terminals without these features cannot be officially called "dumb" for the simple reason that Lear Siegler curiously opted to make the term a trademark and continues to reserve the exclusive right of calling only their own terminals "dumb." Thus we must

use the clumsy term "non-intelligent" to describe terminals without editing features. This is not to be confused with the term "brain damaged," which is reserved for terminals which behave in a painfully undesirable fashion regardless of innate intelligence.

Edit functions are not essential when using **vi**, but they are useful all the same since they can significantly speed editing operations. This is of benefit not only to individual users, whose editing goes much faster, but to the system as a whole because of the lessened I/O requirements for most editing operations.

For instance, **vi** cannot take advantage of the ability of a terminal to perform the *delete line* operation if that capability is not defined in the terminal's **termcap** entry. Let's say you delete the first line on the screen using **vi** on such a terminal; **vi** would then have to redraw the entire screen if it were to try emulating that operation. On most screens, redrawing means that 1920 characters need to be sent to the terminal for just that one operation. At 9600 baud, this takes two seconds to complete, so users obviously have much to gain from a proper **termcap** entry. Less obvious is the waste of system time an improper entry causes, thus slowing system response to *all* users.

Because of this problem, **vi** will mark a deleted line near the top of the screen with '@' rather than redraw the entire screen. However, a *delete line* lower on the screen *will* cause **vi** to redraw the affected portions of the screen (if **vi** is operating at 9600 baud or

higher), meaning there still is some needless system overhead. (At 300 baud, **vi** will use the **@** marker rather than redraw the screen.)

When **vi** was first implemented, intelligent terminals were relatively expensive, so most people had non-intelligent terminals — or even *dumb* (Lear Siegler) models. Now that the costs of intelligent terminals have come down so far that many terminal manufacturers no longer offer non-intelligent CRTs, the problem of supporting additional functionality is almost universal.

For **vi** users, all that is required is to make sure that **termcap** supports the editing functions the terminal offers. Note that some terminals need to be put in a special mode (to be discussed later) to get some intelligent features to work. Generally, this is because those features are undesirable the rest of the time.

When editing features are not available to **vi**, lines and screens are rewritten a character at a time. For this reason, it is often a good idea to debug **termcap** entries at a very slow speed (e.g. 150 baud) to insure you can see what is really happening. The **vi** editor will notice the baud rate and optimize what it does to the screen to match, so something may work at a low baud rate but break at a higher one. *Caveat emptor.*

FAKING IT

If a terminal has an almost-but-not-quite-right operation, it is sometimes possible to get that feature to work by adding cleanup sequences to the appropriate **termcap** field. For instance, if *insert line* (**:al:**) does everything right except that it leaves the cursor on the original line rather than on the new line (the latter being what **vi** expects), then appending an *upline* sequence to the end of the *insert line* description will cause it to leave the cursor where **vi** expects it to be. If the terminal

uses CTRL-T for its *insert line* and CTRL-U for *upline*, then a field like **:al = ^ T ^ U:** would work.

INSERT/DELETE MODE

You can get almost as much mileage out of having just an *insert line* (**:al:**) feature as in having a *full insert mode* (**:im:** and **:ic:**), but both should be specified if your terminal has both.

Note that even if your terminal has an *insert mode* requiring no additional *insert character*



sequences, **vi** still expects at least a null *insert character* field (i.e. **:ic = :**).

Similarly, if a terminal's *insert character* sequence does not need to operate in a special *insert mode*, **vi** still expects at least a *null insert mode begin/end* field (i.e. **:im = :** and **:ei = :**).

Perversely enough, this does not hold true for *delete mode* and *delete character*. If you have *delete character* ability, then you need to specify *delete mode begin/end* (**:dm:** and **:ed:**) only if they are necessary; **vi** does not care. Since it doesn't hurt, it's a good idea to include *delete mode* even if it's left empty (i.e. **:dm = :** and **:ed = :**) for the sake of consistency with *insert mode*.

Be sure to check whether absolute cursor motion works in *insert mode*. If so, specify **:mi:**. On

certain terminals, the cursor motion sequence means something different in *insert mode* than it does otherwise. The **vi** editor needs to use cursor motion in *insert mode* whenever insertions cause a need for screen updates elsewhere on the screen. If **:mi:** is not set, **vi** will need to turn off *insert mode* before doing the other screen updates. This usually does not cost much, but every little bit of optimization helps. As usual, if the terminal documentation does not mention whether this works or not, the easiest way to check is simply to try adding **:mi:** to the **termcap** entry. If strange results occur when characters inserted in the middle of a line cause the line to wrap, then **:mi:** should be removed from the entry.

INSERT NULL vs BLANK

If you define *insert mode/insert character* (**:im:** and **:ic:**), *inserts nulls* (**:in:**) may need to be defined, depending on whether the terminal distinguishes between typed blank spaces and untyped blanks (or nulls) on the screen. After clearing the screen, it will appear to be full of blanks, but the terminal may consider them to be nulls and treat them differently than the typed spaces. Such terminals will treat untyped blanks as "soft" spaces, and will feel free to delete them during insertion operations.

Bill Joy recommends that the following procedure be used with the terminal in local mode (i.e. keystrokes echoed to the screen, but not transmitted to the system) to determine whether the inserts nulls (**:in:**) field should be included in your description:

- 1) Clear the screen
- 2) Type "abc---def" (where each "---" indicates a local cursor motion to the right — that is, use of the right arrow key instead of a space character)
- 3) Use carriage return to posi-



tion the cursor at the start of the same line

- 4) Put the terminal in *insert mode* and type "ghi"
- 5a) If typing the "ghi" does not cause the three blank spaces to be collapsed (that is, if the spaces are preserved and shifted to the right), you do not need **:in:**
- 5b) If typing the "ghi" causes the "abc" to shift over to the "def", thus consuming the blank spaces, you should specify **:in:**

TERMCAP GLOSSARY

The following is an alphabetical list of the **termcap** fields that define the use of a terminal's edit functions:

al - Add Line

Insert new blank line
(string) e.g. **:al = \E ^ R:**

This should open a new blank line *before* the line the cursor is located on, scrolling the rest of the screen down. If the terminal doesn't do it that way, don't specify **:al:**. The cursor should end up at the beginning of the new line. If the terminal leaves the cursor elsewhere, you may be able to fix it by adding cursor motion to the remainder of the **:al:** field. This can be used by **vi** to simulate reverse scroll on terminals that lack an explicit reverse scroll feature.

See **:cs:**, **:sr:**; compare **:dl:**

dc - Delete Character

Delete character

(string) e.g. **:dc = \E ^ A:**

If *delete character* needs a special mode which should not be on during normal editing, it may be turned on via **:dm:** (*delete mode*) and turned off via **:ed:** (*end delete mode*). Otherwise, just **:dc:** is enough. See the discussion under the "Insert/Delete Mode" subheading earlier in this article.

See **:dm:**, **:ed:**; compare **:ic:**

dl - Delete Line

Delete line

(string) e.g. **:dl = \E ^ B:**

This should scroll all lines below the cursor up, eliminating the current line. Don't define **:dl:** if the terminal does *delete line* in a different way. The **vi** editor does not use delete mode (**:dm:**) at all when it uses **:dl:**.

See **:dc:**, **:dm:**

dm - Delete Mode

Enter *delete mode*

(string) e.g. **:dm = \EX:**

If specified, this is sent just prior to any *delete character* sequence



(**:dc:**). You should define *delete mode* if your terminal needs to be in a special mode during *delete character* operations only — that is, if this mode should not be on ALL the time. Only *delete character* (**:dc:**) is used in conjunction with **:dm:** under **vi**, so it is usually not necessary to worry about the finer points of the terminal's behavior while in *delete mode* (for example, **vi** will avoid doing cursor motion while in *delete mode*). Use **:ed:** to specify how to turn the mode off again. See the discussion under the "Insert/Delete Mode" subheading earlier in this article.

See **:ed:**

ed - End Delete

Leave *delete mode*

(string) e.g. **:ed = \EY:**

If *delete mode* (**:dm:**) is defined, *end delete mode* should also be defined so that **vi** will be able to turn it off again. Some terminals treat all cursor motion as a delete operation while in *delete mode*, so problems with this entry will generally be immediately apparent. See discussion under the "Insert/Delete Mode" subheading.

See **:dm:**

ei - End Insert

Insert mode end

(string) e.g. **:ei = \E\200:**

This should be defined any time that *insert mode* is defined. Such **vi** operations as "change word" will cause insertion rather than replacement if this is not correctly defined. See discussion under the "Insert/Delete Mode" subheading.

See also **:im:**, **:ic:**

ic - Insert Character

Insert character

(string) e.g. **:ic = \EI:**

Note that *insert mode* (**:im:** and **:ei:**) must be defined whenever **:ic:** is defined, and vice versa. If putting the terminal in *insert mode* is sufficient to have it insert characters, then define **:ic:** as an empty field: **:ic = :**. An insert character (if defined) will be sent by **vi** before each and every character to be inserted; this is expensive if not necessary. If it is necessary to send a sequence before each character to be inserted, but it is not necessary to be in a special *insert mode* to do this, then define *insert mode* and *end insert mode* as empty fields: **:im = :** and **:ei = :** Some terminals need to be in a special mode in order for insertion to work; specify this mode in **:im:** and **:ei:** only if the mode is inappropriate for other **vi** operations:

In microcomputers today, UniSoft sets the standard.

AT&T has recently been advertising that their UNIX™ operating system will be the standard OS for microcomputers. That's true. But if you want AT&T's UNIX software on micros today, talk to UniSoft Systems.

UniSoft has been delivering AT&T's UNIX adapted for 68000-based microcomputers for two years. More than 75 different computer systems run the UniSoft software, UniPlus+.™ At each Bell release level, all these systems are object code compatible. This means that applications software developed on any UniPlus+ system will work on any other. This is where software portability pays off.

UniSoft enhances Bell's vanilla UNIX with the best features from the Berkeley BSD research version of the UNIX operating system. IP/TCP networking, record and file locking, and virtual memory from UniSoft turn UNIX into a commercial product. All this added value is still Bell-compatible.

Don't wait six months to get System V running on your hardware. UniSoft's customers can ship it now.

If you're building or selling a 68000-based UNIX system, your operating system should come from UniSoft Systems, the UNIX experts.



THE BERKELEY PORT AUTHORITY

Circle No. 16 on Inquiry Card

*UNIX is a trademark of Bell Laboratories

739 Allston Way, Berkeley, CA 94710 • (415) 644-1230

TWX II 910 336-2145 • UUCP ucbvax!unisoft!unisoft

See us at UNIX SYSTEMS EXPO/84, Los Angeles, September 11-14,
Booth #117



otherwise, specify that mode in *visual start* and *visual end* (:vs: and :ve:).

See discussion under the "Insert/Delete Mode" subheading.

See :im:, :ip:, :ei:, compare :dc:, :dm:, :ed:

im - Insert Mode

Begin insert mode

(string) e.g. :|E P:

When possible, this is preferable to using *insert character*. In any case :ic: must be specified (if only as a null field, i.e. :ic=:) when :im: is used. This is incorrectly specified as a boolean field in Chapter 5 of the *UNIX Programmer's Manual*. See discussion

under the "Insert/Delete Mode" subheading and under :ic: (*insert character*).

See :ei:, :ip:, :mi:, :ic:; compare :dm:, :dc:

in - Inserts Null

Insert mode distinguishes nulls on screen

(boolean) e.g. :in:

Some terminals fill empty areas on the screen with nulls after *screen clears* rather than with blanks. Though the two may look the same, they're not. See discussion under the "Insert/Delete Mode" subheading.

See :im:

ip - Insert Pad

Pad needed after inserted

character in *insert mode*

(special string) e.g. :ip=16*:

This represents the amount of delay per inserted character in *insert mode*. It really should be used only for *insert mode* (:im:) since insert character (:ic:) allows delay information to be included in its entry, whereas :im: does not.

See :im:, :ic:

mi - Move Insert

Safe to move while in *insert mode* (boolean) e.g. :mi:

Specify this if absolute cursor addressing works while in *insert mode* so as to prevent vi from turning *insert mode* on and off every time it wants to move the cursor. The vi editor will move the

```
# Concept AVT
#       Simple version with nothing but basic characteristics
#       and editing features
#
c5|avt|HDS concept avt, 80 columns:\
      :is=\E[1*q\E[2\041t\E[7\041t\E[=4;101;103;1191\
\E[=107;118;207h\E)l\E[1Q\EW\E[0\0720\07232\041r\E[w\E2\r\n:\
      :am:xn:bs:li#24:co#80:cl=\E[H\E[J:cd=\E[J:ce=\E[K:\
      :cm=\E[%i%2;%2H:no=\E[H:nd=\E[C:up=\E[A:\
      :al=\E[L:dl=\E[M:dc=\E[P:dm=:ed=:
      :im=\E|:ei=\E|:ic=:mi:\
      :vs=\E[=119h:ve=\E[=119l:
```

Lines 1 through 5 contain the basic capabilities that vi needs in order to work with any terminal. Each of these capabilities is defined in the "UNIX Programmer's Manual" in chapter 5 under "Termcap." Note the :xn: field denoting a "brain damaged" newline: the Concept AVT has automargins, but ignores a newline after it has automatically wrapped the cursor. The :xn: feature was added to **termcap** to support Concept terminals in particular.

Line 6 defines some of the more important editing features: add line (:al:), delete line (:dl:), delete character (:dc:). Note that null definitions were given for delete mode and end delete mode for consistency

with insert mode.

Line 7 continues with insert mode (:im:), end insert mode (:ei:), insert character (:ic:), and move in insert mode (:im:). The Concept AVT uses just insert mode with no insert character sequence, but vi requires a null definition for insert character anyway.

Line 8 defines visual start (:vs:) and visual end (:ve:). The definitions given here cause the cursor to be a flashing block while in vi, and to be a flashing underline after leaving vi. The initialization sequence (:is:) also defines the cursor to be a flashing underline for consistency.

Figure 1 —An annotated sample termcap entry (for a Concept AVT terminal).



EMERALD ONE™

SOFTWARE WITH COURAGE, BRAINS AND HEART

WHAT IS EMERALD ONE?

The most complete integrated office system available today, EMERALD ONE combines the most essential office tasks through six fully compatible and seamless sets of tools. EMERALD ONE runs on a broad range of mainframe, mini, super-micro and personal computers which use the UNIX™ operating system—the emerging standard for the office.

THE TOOLS OF EMERALD ONE

EMERALD ONE integrates your office tasks through:

1. *COMMUNICATIONS*, including Telephone Messaging and Electronic Mail systems,
2. *INFORMATION HANDLING* with EMERALD ONE's powerful Relational Database system,
3. *DECISION SUPPORT* features such as Business Graphics and the Electronic Spreadsheet,
4. *DOCUMENT PREPARATION* with Word Processing and a Cabinet, Document and File Folder system,
5. *TIME MANAGEMENT* tools such as the Personal Diary system and Meeting Scheduler and
6. *SYSTEM ADMINISTRATION* functions that allow a non-technical user to customize EMERALD ONE for the individual, work group and organization with ease.

SOFTWARE FOR THE WORK GROUP

EMERALD ONE goes far beyond stand-alone personal computer software by linking individuals and their work groups. With EMERALD ONE, users work as a communicating group, not as isolated individuals. Whether it be a document, spreadsheet or personal diary entry, everything created with EMERALD ONE can be exchanged easily between individuals, work groups and beyond.

EMERALD CITY, THE PEOPLE BEHIND EMERALD ONE

EMERALD ONE is the result of an intensive, multi-year research commitment by Emerald City and its sister company Trigon Systems Group, one of the most respected consulting companies in office integration.

Attractively priced for distribution by hardware manufacturers, system integrators and OEM's, EMERALD ONE is fully supported by an extensive marketing program designed to assist distributors in penetrating the integrated office market. Emerald City offers the reality of a complete business solution, not just technology.

Emerald City, the company with courage, brains and heart.

EMERALD
C · I · T · Y

Emerald City Inc.
20 Richmond Street East, Suite 700
Toronto, Canada M5C 2R9
(416) 863-9923

See us at UNIX EXPO
in New York City,
Oct. 16-18, Booth #618

UNIX is a trademark of Bell Labs
EMERALD ONE is a trademark of
Emerald City





cursor whenever an insertion causes noninserted characters to scroll or wrap to the next line. See **:im::** compare with **:ms:**

ve - Visual End
screen editor mode end
(string) e.g. **:ve = \EYw\Esl:**

Some terminals need to be put in special 'screen editing' modes to have features such as **:dc:** work at all, but it may be undesirable to leave the terminal in this mode after leaving **vi** or any other screen-oriented program. This can be used to specify what mode the terminal should be returned to after screen editing has ended. Use this to turn off the editing

modes that **:vs:** turns on, if necessary. See **:vs::** compare with **:ti:**, **:te:**

...termcap thus remains a black art.

vs - Visual Start
screen editor mode begin
(string) e.g. **:ve = \EeE:**
Some terminals need to be put in special 'screen editing' modes to

have features such as *delete character* and *insert line*, among others. Use this to put the terminal into appropriate modes for the duration of the **vi** session. See **:ve::** compare with **:ti:**, **:te:**

This article is an adaptation of excerpts taken from Doug Merritt's "Termcap Reference Manual," available from the Independent UNIX Bookstore, 520 Waller Street, San Francisco, CA 94117. Mr. Merritt helped to debug termcap and contributed to the development of vi and curses while attending UC Berkeley in the mid-1970s. He currently works as a consultant out of the offices of International Technical Seminars, Inc., 415/621-6415. ■

JOIN A TEAM ON THE LEADING EDGE OF TECHNOLOGY!

UNIX*, C, PDP 11, M68000

Computer Consoles, Inc. designs, manufactures, markets, and services a variety of minicomputer-based fault-tolerant information systems. Located in Rochester, New York, we have all the cultural and educational advantages of a large metropolitan area as well as the ambience and scenic beauty of a small, relaxed town.

SENIOR SYSTEM SPECIALIST

You will be the lead designer and developer of advanced R&D projects that will extend CCI's Power™ Series of fault-tolerant systems. An advanced degree with extensive experience in UNIX operating system development is required. Additional experience with other operating systems and hardware system design is highly desirable.

SENIOR SOFTWARE ENGINEER

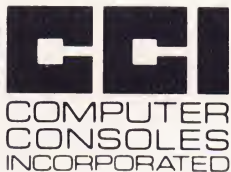
You will be responsible for the design and development of PERPOS™, a UNIX-compatible fault-tolerant operating system on the M68000 and 32-bit super-mini. We prefer a B.S.C.S. plus 3-5 years' related experience. You must have UNIX, "C", and OS internals experience.

GROUP LEADER/SOFTWARE ENGINEERS

You will be responsible for the design and implementation of data base applications and utilities for fault-tolerant operation system development. We prefer a technical degree plus a minimum of 2 years' experience. Knowledge of "C", UNIX and data structures plus experience with data bases in a minicomputer Real Time or on-line environment are pluses.

**UNIX is a trademark of Bell Labs.*

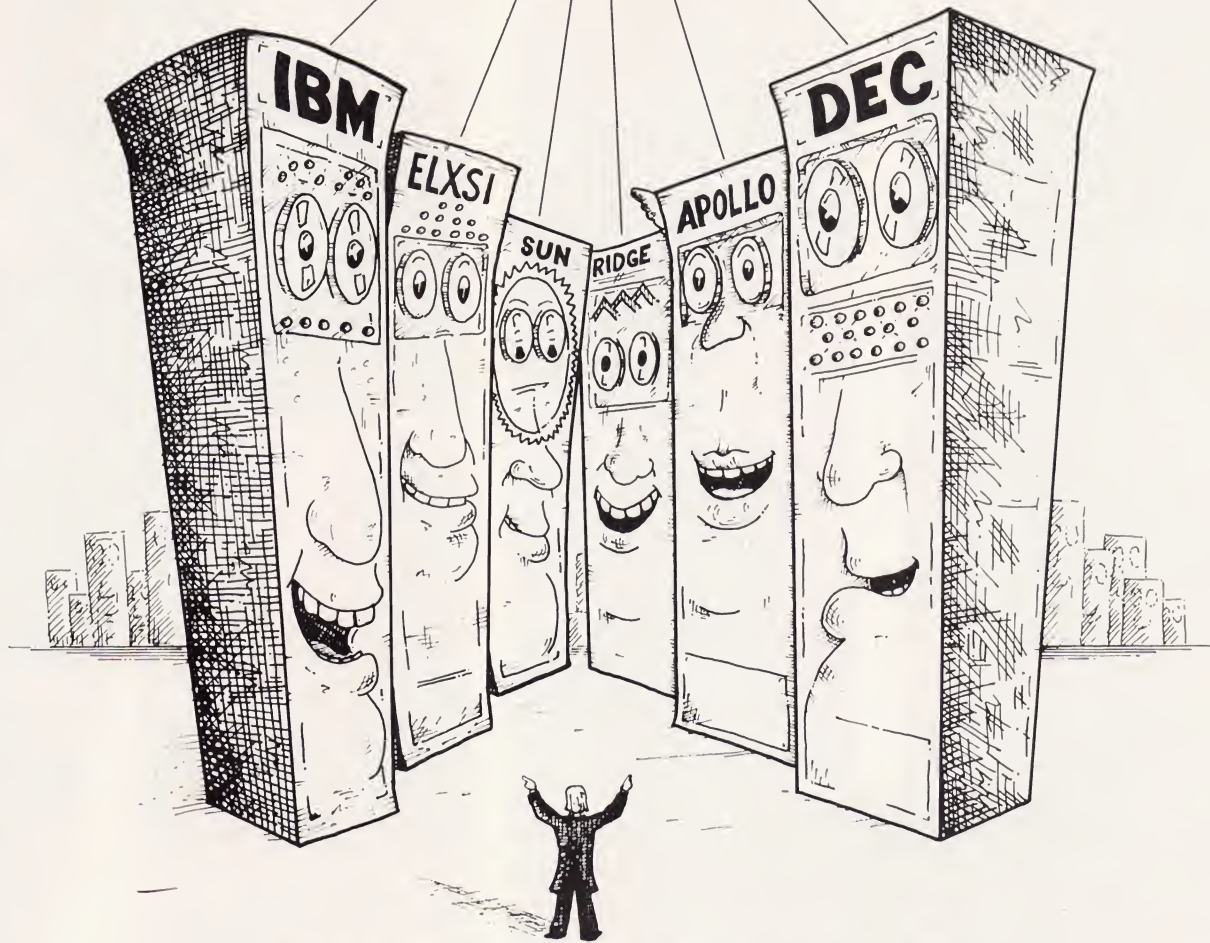
We offer challenging assignments, opportunities for growth, attractive compensation, and a benefits package that includes dental and profit sharing plans. Please send your confidential resume, including salary history and indicating position of interest to:



S. R. Hoskins
Computer Consoles, Inc.
97 Humboldt Street
Rochester, New York 14609
Equal Opportunity Employer M/F/H/V

Circle No. 18 on Inquiry Card

MAINSAIL™



SPEAK MAINSAIL. BECAUSE NOT EVERYONE SPEAKS UNIX.

Most computer programmers like the UNIX environment. It gives them a convenient set of software development tools. There's one problem, however. Applications must often run on machines that don't have UNIX.

That's where MAINSAIL comes in.

MAINSAIL is a powerful programming language that can help cut your development time and eliminate software conversion costs. You'll be able to take advantage of the development power of UNIX, while retaining the ability to move to other systems. And you'll be amazed at how easy it is to learn this proven, versatile language.

If you like UNIX, but need to keep your portability options open, just let us know. We'll show you how MAINSAIL can accelerate your development of portable, sophisticated application programs—not only under UNIX, but under VAX/VMS®, VM/CMS®, and a variety of other operating systems. For details, contact us at: XIDAK, Inc., 530 Oak Grove Ave., Suite 101, Menlo Park, CA 94025, (415) 324-8745.

UNIX is a trademark of Bell Laboratories.
VAX/VMS is a trademark of Digital Equip. Corp.
VM/CMS is a trademark of IBM Corp.

XIDAK

Circle No. 19 on Inquiry Card



UNIX[™] operating systems. An ideal has been

If you've been waiting for an ideal operating system, your wait is over. Now there's HP-UX. It is Hewlett-Packard's enhanced version of the industry-standard UNIX operating systems. And it's available right now on a wide range of HP computer systems.

Yes. It's running on our MC68000-based machines and our powerful 32-bit systems, so

you can pick the right computer for the job.

There are extra features such as graphics and networking. Plus there's a growing array of applications software available for you to take advantage of.

And the HP-UX operating system is backed by our full service organization. As with each of our high-powered systems, we're ready



realized.

to answer questions. Working with both end-users and OEMs, we'll find the best solution for any particular application.

Sound interesting? Call your local HP sales office right now about the HP-UX operating system. Or write to Hewlett-Packard, Attn. Pat Welch, Dept. 100194, 19447 Pruneridge Ave., Cupertino, CA 95014. In Europe, con-

tact Henk van Lammeren, Hewlett-Packard, Nederland B.V., Dept. 100194, P.O. Box 529, 1180 AM Amstelveen, The Netherlands.

Productivity. Not promises.



BDO2421
UNIX is a trademark of AT&T Bell Laboratories.

• SECURITY •

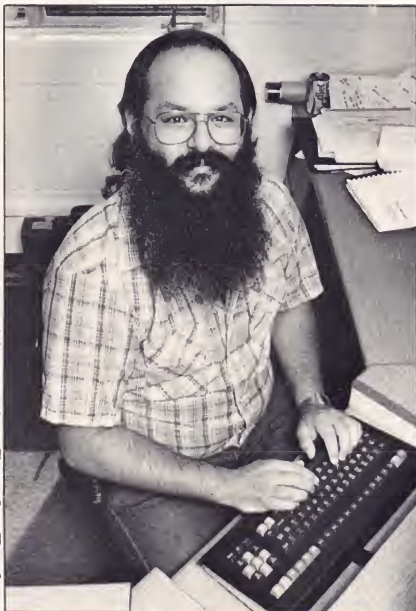
ROUNDTABLE

Experts air their views about UNIX security

A magazine devoted to system administration would hardly be complete without some remarks on system security. Unfortunately, no single approach qualifying as *the* UNIX standard has yet surfaced. To get a grasp on the spectrum of security thinking, *UNIX REVIEW* asked Dick Karpinski, manager of UNIX

services at UC Berkeley to interview a sampling of top authorities. The responses from two, Ed Gould from Mt. Xinu, and Bob Morris of AT&T Bell Laboratories, are printed here. Next month, George Goble of Purdue University and Bob Chancer of Bell Labs will have their say.

Karpinski, a Usenix member of long standing, has frequently been vocal about his own views on computer security. In shaping a series of open-ended questions on the topic, he drew on decades of his own work. The questions were kept essentially unchanged from one interview to the next to allow for easy comparisons.



Photos by Randy Becker

Ed Gould

Many of Ed Gould's views on UNIX security were formed during his years as system administrator at UC Berkeley. Since then, Gould has moved to Mt. Xinu, where he now serves as the leading authority on the workings of the kernel.

REVIEW: *How much security is practical or possible?*

GOULD: Well, what is practical depends on your application. Almost anything is possible. For example, in the dialup case, there are devices that will restrict access by an additional level of password security. You dial up, this box answers and before it will connect the caller, it verifies some password access and perhaps, depending on what level of security you want, calls you back at a number already designated online.

REVIEW: *Those are called port protection devices, I think.*

GOULD: Right, and there are various manufacturers who make those things. To go even further in that same direction, there's complete physical security. Security is always a trade-off between the cost of protection and the value of the data. If your data is sufficiently valuable, then you want to spend a lot to protect it. If your data is not very valuable, then it is not worth spending a lot to protect. You can move up and down that cost/performance curve, depending on your needs.

REVIEW: *Do you see a big distinction between unauthorized disclosure and unauthorized modification?*

GOULD: In terms of what it costs to protect yourself: probably not. In terms of what that does to your system: yes, there is a difference

and it really depends on what you are doing with your data.

REVIEW: *What is the biggest security concern you face?*

GOULD: I guess the biggest concern we face at Mt. Xinu is unauthorized disclosure of data that we are contractually obligated to protect — in particular, the UNIX source. Beyond that, we have our own business data online but that is probably not valuable to too many other people.

REVIEW: *What measures do you recommend for dealing with these concerns?*

GOULD: Generally, I recommend that people use the UNIX file protection system in a fairly full way — that they decide at the outset what kind of default protections they want for the file systems they create. I also advise them to be aware that they may want to restrict access to some files or grant additional access to others. From a system administrator's point of view, they also need to be aware that individual users will need to have their own personal **userid**s so that they can be private when they want to be, but that they also need access to the shared files of whatever group they're working with. Basically, all users should be educated about what they are taking or giving away when they set the permissions. Generally, users will be responsible if they are shown how to protect themselves.

REVIEW: *Do you recommend anything beyond taking a look every once in a while to see how permissions are set?*

GOULD: No, I think that they are worth setting up right when you first build the system. Maybe then you make sure they stay right every now and then, but that doesn't tend to change unless you get broken into.

The UNIX protection scheme from sort of an academic, theoretical point of view is pretty good. It provides a fairly tight security model that is reasonably flexible. The biggest problem that it has is that the people who

distribute UNIX systems operate in very open research environments, so they distribute systems with permissions set in a very permissive way. When people get these systems, they don't often go through to reset the permissions to what is appropriate for their installations.

REVIEW: *How do you measure the costs of the security measures you employ?*

GOULD: Some of them are hard to measure. Some are easy. The hardware devices, of course, have

All users should be educated about what they are taking or giving away when they set the permissions.

a dollar price. But the intangible prices of protections are determined by how much of a nuisance they are to users.

REVIEW: *Somebody pointed out to me that if an ordinary user without superuser privileges runs into permissions that are set wrong, it will generally take that person a day to find an appropriate person to set the permissions right.*

GOULD: Yes, I'm sure that is true in many situations. But that brings up one more thing that I would encourage people to do to protect themselves: never do anything as superuser that can be done in a different way. Superuser status is just too powerful. From two points, it is dangerous. One is that people make mistakes, so you want the user protection system to protect yourself from your own mistakes. If you make a mistake as superuser, the system won't try

to do anything to protect you. That is the biggest reason, but obviously there are also things that the superuser can do that you don't necessarily want ordinary users to be doing routinely.

When you have superuser permissions, you should be thinking very clearly about what you are doing and why you are doing it because you are violating the system's protection scheme and you'd best have a reason for it. It's not good to get in the habit of doing things as superuser. Sometimes it's a little easier to do it that way, but it's a dangerous pattern to fall into.

REVIEW: *As we go through time in a given installation running UNIX, would you expect that those things that are found to be common superuser activities would themselves become encapsulated in programs making minimal use of superuser powers?*

GOULD: Yes, that is usually a good idea. A lot of people don't tend to do that, though. There are some fairly routine things that they will continue to do as superuser. Backups tend to be one of those. But it is not necessary. For example, a reasonable way to do file system dumps is to set someone up with an operator account that has read permissions to the disks. Of course, you don't normally want users to have read permissions to the disks because that in itself is a security violation.

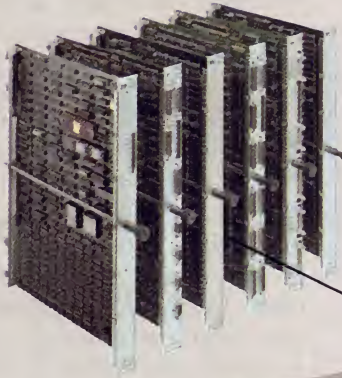
REVIEW: *How much security is enough?*

GOULD: That again depends on the application. It is a price/security trade-off. The more security you have, the more difficult it often is to do things, so you need to trade that off against the value of your data.

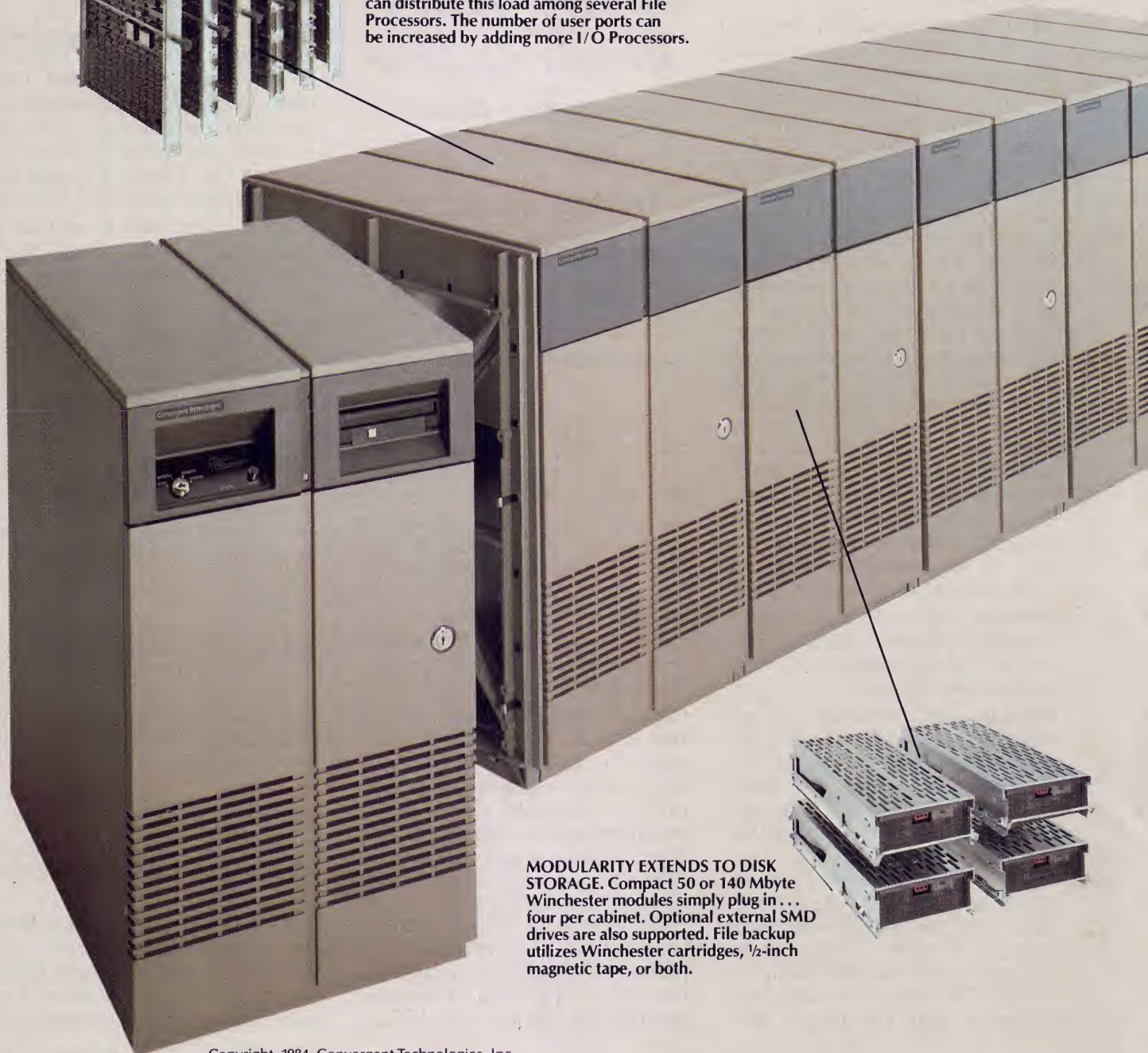
REVIEW: *So you are really starting with the question of how much a breach of security would cost?*

GOULD: Right. Sometimes security is like insurance. You don't want to buy more than you need, but you probably need some. The more you get, the more it costs.

IT WILL GROW ON YOU.



CONFIGURABLE ACCORDING TO USER NEEDS. As many as six processors can be installed in each enclosure. CPU-intensive jobs utilize multiple Applications Processors. Systems with heavy disk usage can distribute this load among several File Processors. The number of user ports can be increased by adding more I/O Processors.



MODULARITY EXTENDS TO DISK STORAGE. Compact 50 or 140 Mbyte Winchester modules simply plug in . . . four per cabinet. Optional external SMD drives are also supported. File backup utilizes Winchester cartridges, 1/2-inch magnetic tape, or both.

MegaFrame™. Now OEMs can offer a high-performance UNIX™-based system that can't run out of performance.

OEMs can now deal cost-effectively with the problems encountered when user applications produce computing demands that outstrip the capabilities of conventional systems.

Convergent Technologies' MegaFrame is a revolutionary new UNIX-based super-minicomputer—so innovative in its architecture that it represents the ultimate in multiuser systems design. It grows exponentially from a system offering minicomputer-level performance to an enormously powerful engine serving as many as 128 users with 36 parallel processors, 24 megabytes of RAM and gigabytes of disk storage.

No other system can match the MegaFrame's potential for field expansion. It enables manufacturers and systems builders to keep pace with today's requirements for more and more computing services... but *not* at the cost of discarding hardware or performing expensive CPU upgrades.

MegaFrame's architectural breakthrough. Dependence on traditional single-CPU shared-logic architecture is the root of systems bottlenecks.

Convergent's response: a novel system utilizing *multiple* specialized processors to distribute workloads for optimum performance—even if user needs are unpredictable or subject to rapid change.

MegaFrame's virtual memory Applications Processors each have a 32-bit CPU, up to 4 Mbytes of RAM and run a demand-paged version of UNIX System V. Up to 16 of them can operate in parallel.

The File Processors effectively function as back-end machines providing DBMS, ISAM and other disk-related services. Up to six File Processors each with four disks can operate in parallel.

Terminal and Cluster Processors can also be added—the latter serving front-end communications needs. They off-load communications from the other processors by running protocols such as SNA and X25 networks.

MegaFrame's daisy-chained cabinets offer total expansion potential of up to 36 slots. OEMs configure the system needed for specific applications simply by adding the correct number/combination of processors.

Flexibility in applications development. Inclusion of one or more Applications Processors allows running UNIX System V. All standard UNIX tools are provided, along with COBOL, FORTRAN-77, BASIC interpreter and compiler, plus Pascal.

The "least-cost solution" to serving a wide range of UNIX-systems needs, MegaFrame has won acceptance from OEMs in the U.S. and abroad. The uniqueness of its modular design, its versatility in providing upgrade-path options and its price/performance advantages give it market-share potential of outstanding dimensions.

The system that will grow on you starts at a very attractive price: about \$20,000 for a system that effectively supports 16 users. Send now for a comprehensive Information Package including reprints of magazine articles. It explains how MegaFrame's growth potential can impact favorably on your plans for growth in the UNIX market.

Convergent Technologies, Data Systems Division, 3055 Patrick Henry Drive, Santa Clara, CA 95050. Phone: 408/980-0850. Telex: 176-825.



MiniFrame™: the entry-level multiuser UNIX system.

Starting at under \$5,000 for a single-user system, Convergent's MiniFrame offers outstanding capabilities for small to medium sized organizations running large UNIX-based applications. Utilizing an MC68010 microprocessor operating at 10Mhz, with no wait states, it provides impressive CPU speed—comparable to VAX™-11/750 running the AIM™ Benchmark. MiniFrame features virtual memory management, with demand-paged implementation of UNIX System V. It runs as many as eight terminals, with up to 50 Mbytes of integral mass storage. MiniFrame and MegaFrame are object-code compatible, allowing OEMs to offer a *complete family* of systems unrivaled in price/performance characteristics.



Convergent Technologies

Where great ideas come together

MiniFrame and MegaFrame are trademarks of Convergent Technologies, Inc. UNIX is a trademark of Bell Telephone Laboratories, Inc., VAX is a trademark of Digital Equipment Corp.



REVIEW: How would you know if you have too much?

GOULD: It is hard to know if you had too much. You find that you had too little if you get broken into and you learn that you have too much when it impedes everybody's work.

REVIEW: Have you experienced any security breaches?

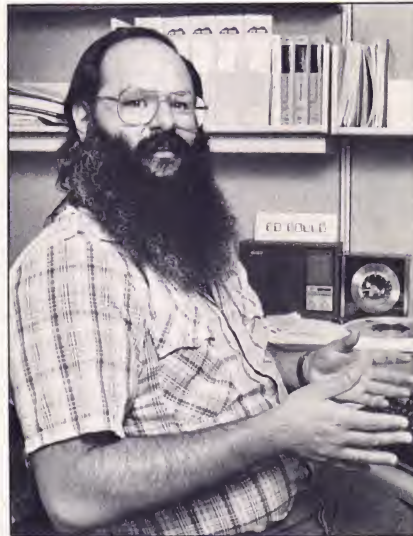
GOULD: I certainly did at the university, but I haven't here. At UC, we had someone who broke into the system more or less as a game to see if it could be done and then decided to be a good Samaritan and install the latest version of whatever it was that he could get his hands on. He wasn't trying to be malicious, he was just trying to leave his mark on things to make sure people knew he had been there.

REVIEW: Do you know of any security breaches that were in fact malicious?

GOULD: I've never had any personal experience with that.

REVIEW: What do you see as the impact of installing UUCP?

GOULD: UUCP is a potential security hole. Typically, as it's distributed, it lets people copy anything into or out of your system within the realm of the file system permissions. Many systems allow almost everyone to



read most files — especially the binaries and in some cases, the sources of things. It is fairly easy to restrict UUCP to its own spooling directory, and many people do that as soon as they get it, but it is distributed in a fairly open form.

People have suggested that the right way to do things is to only allow users to initiate file copies outwards from their own machines — never allowing outside users to initiate outgoing copies. That has been suggested but not widely implemented.

REVIEW: How about the impact of local area networks on system security?

The intangible prices of protections are determined by how much of a nuisance they are to users.

GOULD: It depends on what you think of as the system. If you think of it as one CPU, then a local area network has a fairly severe impact. If you think of it as the whole collection of machines on the network, then it has very little impact because you don't really care what happens between the subpieces.

REVIEW: But if there is an untrusted machine that could be a promiscuous listener, then it might...

GOULD: Yeah, if you have untrusted machines on the network, then that's a problem. A promiscuous listener can get at any of the data that transfers on the net. The only real way to deal with that is an encryption scheme, which as far as I know is offered by none of the existing networks, except perhaps as an option that the user can invoke before submitting any data.

REVIEW: That leads right into my next question. What do you see as the future of security management?

GOULD: Well, it's always going to be around. People are always going to be concerned with their data, either because they have

Image Network's XROFF, right now, prints troff/ditroff documents on:

- VAX
- Pyramid
- Plexus
- PDP 11's
- Integrated Solutions
- Amdahl
- IBM-PC
- 3B20
- System 5
- System III
- Berkeley 4.2
- IS/WB
- V 7
- UTS
- MS/DOS
- Xenix
- Xerox 2700
- Xerox 8700
- Xerox 9700
- Diablo ink jet
- Diablo thermal
- Dec LNO1s
- Compugraphic 8400
- APS-5 typesetter

at leading organizations from Berkeley to Murray Hill.

Call or write with your requirements!
Image Network, 770 Mahogany Lane, Sunnyvale CA 94086
(408)746-3754

This ad was set using Xroff on a Xerox 2700 laserprinter

Circle No. 80 on Inquiry Card

contractual obligations to protect it or because it's proprietary and they want to protect their place in the market. We're certainly not headed for a society where everybody shares everything with everyone else. If we were, then perhaps security issues would go away, but we're not. I don't see anyone developing any new kinds of security, though.

REVIEW: *But you mentioned encryption a minute ago. That's something that nobody's doing now. Do you see that happening at a later time?*

GOULD: Yeah. Encryption hardware is going to become part of networking systems at some point. There has been a lot of controversy about the existing encryption schemes. Many people believe that they are not adequate for reasonable protection and so aren't even worth the very small price they usually cost.

REVIEW: *This is talking about the DES (Data Encryption Standard)?*

GOULD: Yeah, the existing DES is believed by many people to be a wholly inadequate encryption mechanism. The key is from a quarter to a half the size that it might reasonably be. The possibility of Trojan Horses in the encryption scheme has been raised many times. The current DES hasn't really been jumped on. But on the other hand, the financial community is encrypting the data it sends to automatic teller machines.

REVIEW: *What about "capability" schemes?*

GOULD: Most capability schemes have been fairly inefficient. One of the questions that tends to come up in the design of a protection scheme is: how does one revoke

permission? One wants to be able to give permission to someone to access something and then be able to revoke it later if necessary. Typically, a capability is something you hand off to the user. It says, "Holding this thing gives access to that object." It's a key, but how do you re-key the lock?

All the protection schemes that have been discussed have good points and bad points in areas like that. But they're still active. There's a lot of research being done in that sort of thing.

REVIEW: *What of the move of UNIX into the business marketplace? With it showing up on everybody's desk, do you see a big education problem coming?*

GOULD: Yeah, and I think that's exactly the hard part of providing really good security. The number of people who now have to be educated about it is growing at an immense rate. Some installations may not want to put that responsibility for security on each individual because any individual who doesn't choose to comply might be able to compromise the entire system.

REVIEW: *Does this mean more*

courses, more seminars, more in-house training?

GOULD: It means more inhouse training with a particular emphasis on understanding what the existing protection scheme is and how to use it. That's something that seems to be left out a lot now. People want to learn how to use the tools first. They want to learn how to edit their document or write their program or whatever. Security tends to be a later concern. It probably should be an early concern.

REVIEW: *Perhaps more splashy press coverage is necessary. Maybe the publicized break-ins accomplish some good.*

GOULD: There are two reasons to not consider people who break into systems as heroes. One is that what they've done is not very difficult. Also, even though they might ultimately be performing a service by making people tighten up their security, I don't think it's reasonable or appropriate to glorify that sort of activity. We wouldn't encourage people to go around breaking into houses in order to convince people they should put better locks on their doors — even if all they did was walk up, open the door and leave.

Continued to Page 60

YOU can print your
XROFF manuals, proposals, forms...

right in your own shop!

Use **your** computer and our software (**XROFF**) and fonts (we have 100's) with the laser printer, typesetter, inkjet or dot matrix printer of your choice. We can provide whatever equipment you don't already have, at a price less than you would think.

Call or write for more information:

Image Network

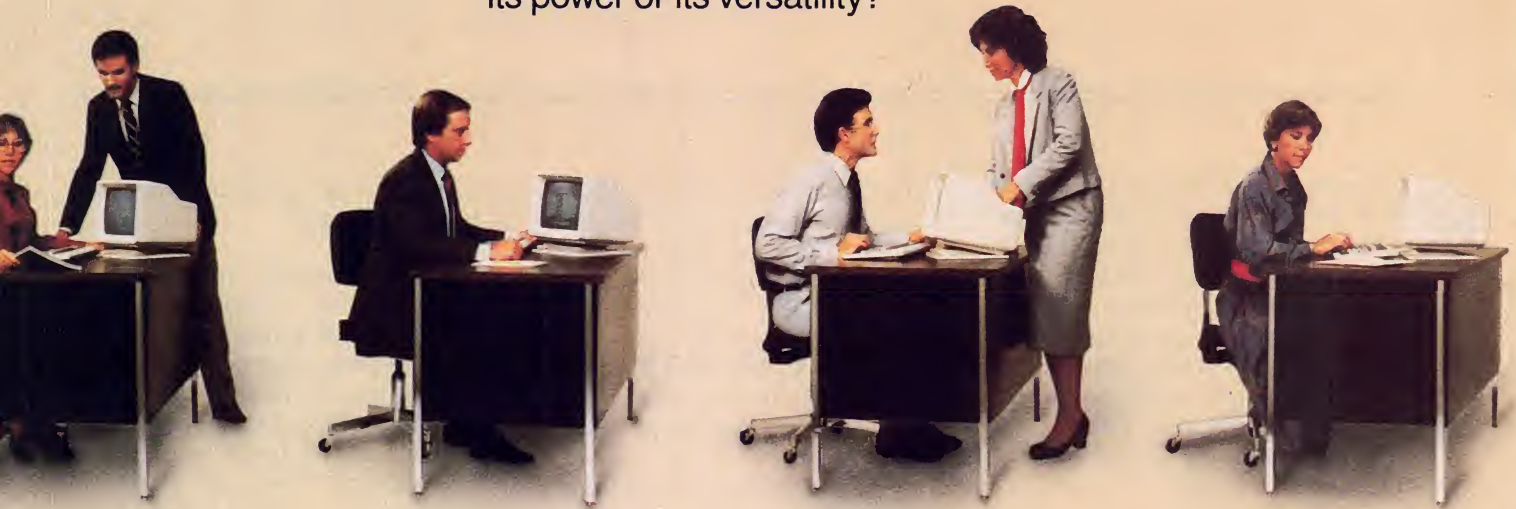
770 Mahogany Lane, Sunnyvale CA 94806 (408) 746-3754

This ad was set using XROFF on a Xerox 2700 laser printer

Circle No. 80 on Inquiry Card

What do you like
better about the VISUAL 2000 ...
its power or its versatility?

Personally,
I like its price!



Never has a UNIX-based multi-user system given so much to so many for so little.

Introducing the VISUAL 2000

The VISUAL 2000 is the full-featured system with the power and flexibility to support multiple users in real business applications at a surprisingly low cost per station. It can be used with inexpensive video terminals. Or as a database manager or file server for a cluster of intelligent workstations or PCs, including both the IBM® PC and VISUAL's own lightweight, portable, totally IBM PC compatible COMMUTER. In all applications it offers greater performance, more flexibility, and lower cost than any other system in its class.

Powerful Intel 286 processor

The Intel 286 is today's chip of choice for UNIX™-based systems. Only the Intel 286 has on-chip memory management, an instruction set optimized for multi-tasking, and the optional 287 numeric co-processor to speed up floating point by a factor of 10.

What do these features mean to the VISUAL 2000 end user? Faster response time, more users supported, and lower system cost!

Cost-effective one-board design

A basic advantage of the VISUAL 2000 is its one-board base-level design. A single high-density board includes the 286 CPU, 512KB-2MB of RAM, controllers for Winchester, floppy, and streaming tape, an intelligent communications processor, six RS-232 ports, and a parallel printer port. Even a real-time clock with battery backup. One-board design means higher performance, lower cost, and greater reliability than comparable multi-board implementations.



Configurability and Expandability: VISUAL gives you more

The VISUAL 2000 spans a much wider range of configurability and expandability than other systems in its price class. Up to 16 independent users. 6 megabytes of RAM. 4 Winchester. Floppy. And streaming tape for simple, reliable disk backup. All in a small stand-up enclosure which looks right at home next to a desk.

And if a fully expanded VISUAL 2000 isn't enough, you can connect up to 254 VISUAL 2000s, PCs, and workstations in a local area network.

Extensive system software simplifies system integration

The VISUAL 2000 runs XENIX, Microsoft's popular, enhanced version of UNIX, derived from UNIX under license from AT&T, and designed to be faster, more secure, and easier to use in business applications.

And VISUAL has worked hard to simplify the system integrator's job, by providing all the tools needed to deliver end-user applications with a minimum of effort.

Languages such as C, SMC BASIC, RM/COBOL, TOM BASIC, SOFTBOL, and MicroFocus Level II COBOL, to provide instant compatibility with hundreds of proven business application programs.

Other system-building tools, like the INFORMIX database management system and RealWorld modular accounting system.

And productivity software, such as the 20/20 integrated spreadsheet and XED office-grade word processor.

The Bottom Line

High performance. Superior flexibility. Extensive software. And low cost... VISUAL 2000 systems start at under \$10,000, suggested list. No one gives you more in a UNIX-based multi-user system.

Whether you're an OEM, system house, distributor, or end-user, call today for further information on the VISUAL 2000 and see for yourself!



VISUAL See for yourself®

Visual Technology Incorporated
540 Main Street, Tewksbury, MA 01876
Telephone (617) 851-5000. Telex 951-539

REGIONAL OFFICES:

Northwest:	(415) 490-1482
Southwest:	(213) 534-0200
North Central:	(513) 435-7044
South Central:	(214) 255-8538
Northeast:	(201) 528-8633
Southeast:	(301) 924-5330

Circle No. 22 on Inquiry Card



Bob Morris is an AT&T Bell Laboratories mathematician noted for his research in computer security. He currently supervises Bell Lab's Signal Processor Systems Engineering Division in Whippany, NJ.

REVIEW: *How much security is practical or possible, and ultimately at what cost?*

MORRIS: Well, I think a great deal is possible. I certainly know of a number of places where I couldn't possibly break in, steal information, get unauthorized access, plant Trojan Horses or do anything of the sort. They're just too good for that.

REVIEW: *How do they do that?*

MORRIS: One way is to disconnect all the telephone lines. You can put the computer in a shielded room and put a guard at the door.

For example, several of our country's intelligence agencies use UNIX on a timesharing basis. They have no dial in lines, and the computer is guarded. You can't get at it.

Many large companies protect their payroll, Accounts Receivable, Accounts Payable and Personnel files in exactly the same manner.

REVIEW: *What about leased line access? That, of course, is subject to taps if the line goes out of a physically secure area.*

MORRIS: In all of the cases I know about, that's true. All classified information that goes outside a secure area must be protected either by cryptography or a gas-filled line. That's universal in the classified information area.

REVIEW: *Now, when you do that, what are the costs?*

MORRIS: I don't think a pressurized line costs more than ten to twenty dollars a foot. It's hard to say exactly what the cost is

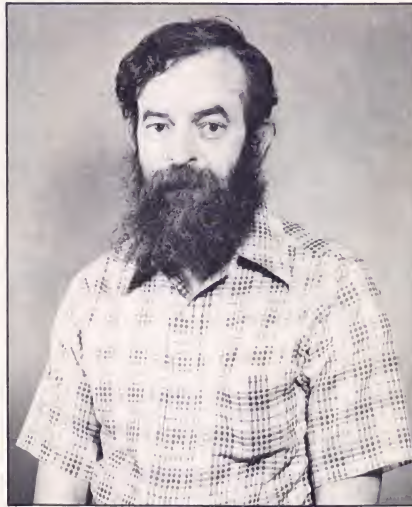


Photo courtesy of AT&T Bell Laboratories.

Bob Morris

because another cost is preventing or making it difficult or expensive for *authorized* people to get access. For example, at an intelligence agency, you don't expect that your employees will be working on their files throughout the evening. It's also quite difficult and expensive to transfer information from one agency to another.

There is both an investment cost and an operating cost. My attitude is that computer and information security is primarily an economic problem and that the keys to it are almost completely administrative, not technical.

They don't have to do with cryptography or jazzy new programs or anything like that. They mostly have to do with administrative tradeoff studies of setting and enforcing policies. Someone has to sit down and decide these cost tradeoffs. Once that's done, the administrators will at least have the guidelines they need to institute and enforce the policies.

Now there are systems that can be made secure to a very high degree. For example, there are systems that will not accept logins

by incoming telephone calls. If a caller says who he is, the computer will dial back a telephone number already stored in memory that corresponds to the person's name.

Another technique which is actually used is cryptography, which is very expensive.

REVIEW: *That's expensive, but presumably the advent of silicon implementations of serious encryption algorithms will make that cheaper.*

MORRIS: Oh, sure, but I'm not talking about chip cost. I'm never talking about chip cost. I'm talking about administrative cost. How do you distribute cryptographic keys? How do you generate them? How do you protect them? The cost is administrative.

Understand that there are levels of security that are amenable. You measure the cost and you decide what level you want for your information. To do this job, you assess the exposure: what's the nature of the information, and what's the risk? Take for example a company like an international pharmaceutical firm, where the director's office necessarily assumes a personal criminal responsibility for safeguarding much of the information about the company's international operations. There are substantial penalties under this. There are also financial risks for the company if its information is stolen by competitors.

REVIEW: *What's the biggest security concern or problem that you face?*

MORRIS: Inadequate administrative control.

REVIEW: *What measures do you recommend for these concerns?*

MORRIS: I think you could guess my answer. Well, let me tell you. First, an awareness of the problem on the part of top management,

Continued to Page 98



WHAT YOU DON'T KNOW ABOUT UNIX* CAN STUNT YOUR GROWTH.

If your business depends upon computers and you *do not* know how UNIX will allow it to grow, your computer system could be threatened by obsolescence.

However, if you *do* know about the explosive emergence of UNIX as an industry standard for business, engineering, and scientific computers, as well as a multi-user and multi-tasking environment, you'll stand to reap extraordinary user benefits.

Benefits that free you from the tyranny of dependence on a single hardware manufacturer. Protect your investments in software when you want to upgrade hardware. In short, the benefits of UNIX will grant you a freedom of choice that you never had before.

UNIX SYSTEMS EXPO/84 offers an unprecedented opportunity for business managers, information systems professionals, and all others with a need to better understand the full impact of UNIX. Over 150 of the industry's leading and most innovative companies will be exhibiting the latest hardware and software products. More than forty informative sessions will be offered and oriented to issues that are of prime concern to current and prospective users of UNIX-based systems.

The choice is yours. You can be content with your understanding of computer

technology as it is. Or you can learn everything you need to know about UNIX. And *grow*.

Early Bird Registration will be held Monday, September 10, from 3:00 p.m. to 8:00 p.m. Or register during show hours:

Tuesday, September 11,

11:00 a.m. to 6:00 p.m.

Wednesday, September 12,

10:00 a.m. to 6:00 p.m.

Thursday, September 13,

10:00 a.m. to 6:00 p.m.

Friday, September 14,

10:00 a.m. to 5:00 p.m.

For information, call (East Coast Office) 617-965-8350 or (West Coast Office) 415-364-4294.

UNIX* SYSTEMS EXPO/84

September 11-14, 1984 Los Angeles Convention Center

An exclusive production of Computer Faire, Inc.
A Prentice-Hall Company

*UNIX is a trademark of AT&T Bell Laboratories

Circle No. 23 on Inquiry Card



HORROR STORIES

Remembrances that confirm Murphy was an optimist

by Dr. Greg Chesson

Two days before the Big Deadline, the programmers began to stir. One day later, they got serious, as programmers always do on the Day Before. Little did they know they were headed for the Twilight Zone of computer operations: the Power Outage. On this occasion, the building electricians 'lost' one phase of computer room power while working on a distribution panel. Since the computer systems were protected by a power monitor, the accident should have caused an immediate system shutdown.

But the power monitor had recently been cutting off the computer systems for no apparent reason and the fail-safe controls on the power system had been disabled by the computer vendor's field engineer. So instead of a clean computer shutdown, there was a mild disaster. One of the disk drives, the one with THE important data, was left scrambled by the operating system, and some memory boards developed parity errors when the power came back on. But, perhaps worst

of all, the cappuccino machine was unusable while electrical work continued.

You probably know what happened next. Programmers and documenters alike left the building to find cappuccino and fret about the Big Deadline. The system administrator, meanwhile, stayed behind to put things back together and answer phone calls from the programmers.

This kind of story should be as familiar as Murphy's Law to anyone who has ever used a computer system and had a production deadline. As system horror stories go, this one is mild. But it illustrates the basic ingredient of unexpected — although perhaps avoidable — computer failure mixed with a little black humor.

GREMLINS AT WORK

System disasters frequently involve disk drives. Classic stories often recount the moving of a crashed pack to a good drive, or the use of good packs on damaged drives, or some combination thereof. More mysterious than the disk crash is the disk system that



acts as though it has had a head crash but is in fact all right. One symptom of a minor head crash is gradual loss of the ability to read or write portions of the disk surface. Ponder for a moment your disks that appear to be suffering head crashes, but are in fact simply losing data. This very thing happened to me last year.

We bought some new disk packs for a 300 MB disk system that were expensive and guaranteed to be error-free at the bit level. After formatting and testing the packs, it seemed the claims of the manufacturer were correct. But after about two weeks use, we started to get header errors on one of the new packs. Fortunately, there were good tape backups and spare disk packs. After the computer vendor's field engineer inspected the drive and declared it in good health, we were ready to forge ahead with another new pack. But about two weeks later, the problem returned and the same recovery procedure was followed. This time the field engineer gave us some static about not using disk packs purchased from his company. This was especially irritating since we knew the packs we were buying were of better quality than the ones we could get from his company.

After about two months of continually losing data in this way, we were reasonably sure something was wrong with the disk drive. We were able to see a pattern in the errors: the disappearing data clustered under the same head. Mysteriously, though, the head was normal. Then we noticed that the failing disk sectors were all related to files that had not been accessed for about two weeks prior to the disk "crash."

After appeals and sacrifices to the field engineering gods, we conveyed this new information to a

field engineering supervisor who really understood this piece of equipment. A long evening with test equipment and diagnostics led him to the discovery that a failing diode on a current-switching card was applying a very light write current *at all times* to one of the disk heads. Everything underneath this particular head was subject to being erased, with

**If a new system
caretaker cannot
find all the legacies
that have been left
behind, they will
announce
themselves soon
enough.**

about two weeks being necessary to complete the job. Only frequently written disk blocks were immune to the gradual erasure process.

This story has the elements of a "classic" because it combines an obviously failing piece of equipment with the woeful inadequacy of system diagnostics. Programmers and system administrators alike cringed in the certain knowledge that every two weeks they would have to go through the painful process of restoring data from tapes. The process was repeated over several months, distinguishing it from the two tales of unique system crashes I'm about to unfold.

TABLE AND CHAIR SHUTDOWNS

We once had a cheap disk controller that was a thin, flat box mounted in a conventional 19-inch rack on sliding rails. Cables plugged into the rear of the controller led to the disks. The controller was located at about waist height for easy access.

One afternoon UNIX suddenly stopped on the system this controller operated. We trooped into the computer room to see what had happened and discovered that one of the gang had pulled the controller out of the rack because it looked like a useful table. The disk cables fell off the rear of the box — there was no strain relief — and the file system slowed by quite a bit. We fastened things down a little better after that.

On another occasion, I was in a computer room, sitting at a console while writing some tapes. There was but one chair in the room. And so, when a colleague came into the room to chat, he began to lean against the mainframe, watching what I was typing as we talked. He gradually began to "sit" lower and lower until his head was about at the same level as mine. At about this time, UNIX stopped. My friend had sat on the halt/start switches of a DEC 11/70.

THE DIAGNOSTIC BLUES

In the early days of UNIX, field engineers would assume that an unattended machine was idle. This was a natural assumption because in those days, timesharing on a minicomputer was a new concept. Far too often, a field engineer would walk up to a supposedly idle CPU, load a diagnostic tape and halt the machine to begin running diagnostics. Shortly thereafter, he would be basking in the consternation of a swarm of

angry programmers. Times have changed, but the problems remain.

We once had a computer room full of equipment from manufacturer A. Not being completely satisfied with what vendor A was offering, we had also purchased a machine from vendor B for research and evaluation. The sales and maintenance people from vendor A derived pleasure from any obstacles they could place in the path of vendor B. This was perhaps best illustrated one morning when a field engineer from vendor A working in our machine room noticed a malfunctioning memory board in the B machine begin to heat and generate black smoke. He waited until it was pretty well smoked before coming out of the machine room to announce, "I think your other machine is on fire." He was just trying to help.

FIRE, FIRE!

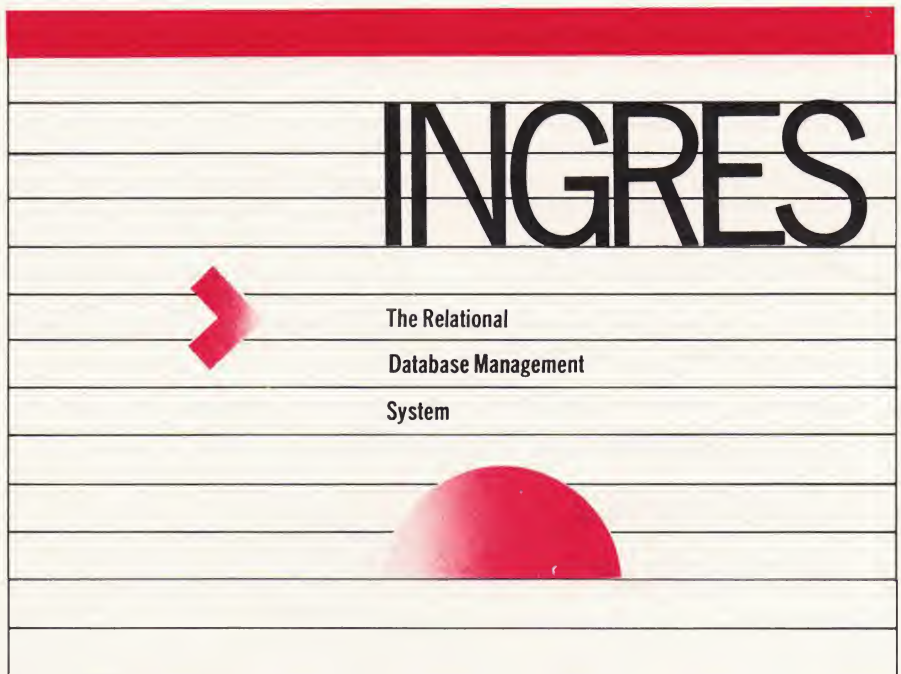
Stories of computer equipment burning are rare, so here's one that is a personal favorite because it involves a legacy I left behind in graduate school. "Legacy" in this case refers to a hardware or software item that has been around so long that its presence is not generally noticed or even known to its heirs.

As it happened we had a UNIX machine with 16 asynchronous ports and no funds for expansion. I had an idea for doubling the number of asynchronous terminal lines on our machine that involved making little circuit cards that would use one bit of a byte on a serial line to submultiplex the line. Each circuit card had three terminal connectors: one went to the computer and the other two each went to a terminal. An open frame chassis holding 16 little circuit cards was constructed and placed under the raised flooring in a

remote area of the computer facility. The existing device driver was fiddled with to handle the extra terminals, and in due course we had a 32-port system.

This system was still working

several years after the people who built and installed it had left. But then came one fateful day when somebody's terminal didn't work. In the course of finding out why, the cable continuity between the




INGRES

The Relational Database Management System

- Powerful/Easy to Use
- Visual Programming Capability with:
 - Application Generator
 - Integrated Report Writer
 - Integrated Business Graphics
- Integrated Data Dictionary
- Non-Procedural Query Language
- Exclusive Networking Capability
- Supported Around the Clock

AT&T INGRES is now available on the 3B5 and AT&T INGRES/CS on the 3B2!

VAX and VMS are trademarks of Digital Equipment Corp.
MC68000 is a trademark of Motorola Corp.
UNIX is a trademark of Bell Laboratories.

 Relational Technology Inc.
2855 Telegraph Avenue
Berkeley, CA 94705
415-845-1700

Circle No. 24 on Inquiry Card

See us at UNIX SYSTEMS EXPO/84, Los Angeles, September 12-14, Booth #150 and at UNIX EXPO, New York, October 16-18, Booth #312



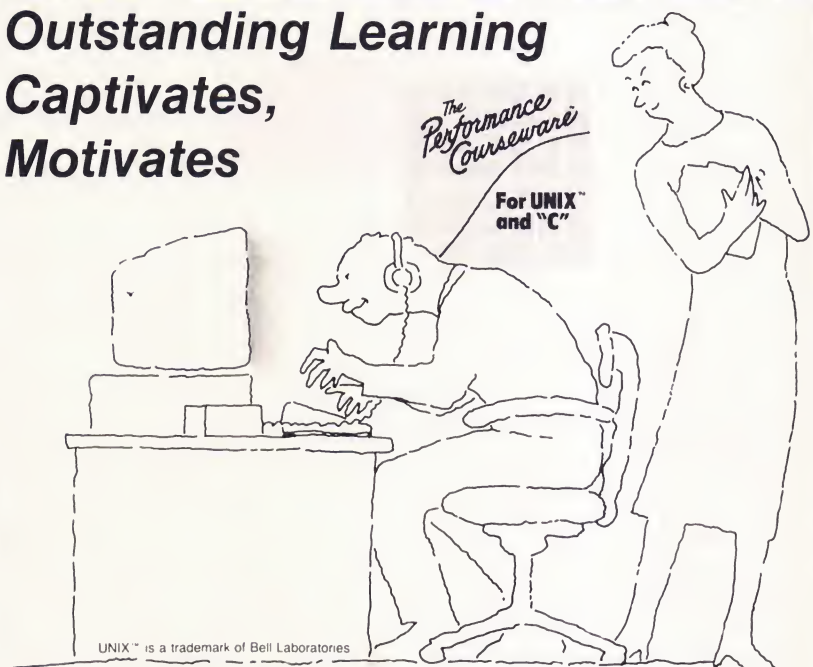
terminal and computer room was tested. Of course the cable for the unhappy terminal could not be identified from among the 16 leading into the computer. So at some point during the discovery process, there was a great deal of pulling and yanking of cables. In the process, the open frame circuit card assembly and its power supply were apparently put into contact with metal floor supports. By the time our forgotten assembly was pulled from the depths, it had been reduced to a gooey, smoking mess. I heard my name was actually remembered that day.

One moral to this story is that if a new system caretaker cannot

find all the legacies that have been left behind, they will announce themselves soon enough. Another observation about all the stories offered here is that they are based on "old" technology. But don't think for a moment that new technology is free of problems. I recall the confusion that resulted when our Ethernet cable was unknowingly severed when a tap was installed. The echoes from the unterminated line crashed some computers, leading us to think for a while that something was wrong with our machines. The real story here is that complex systems and complex humans will continue to provide grist for the horror story mill.

Dr. Greg Chesson serves on the Editorial Review Board of UNIX REVIEW and is presently a Senior Scientist at Silicon Graphics, Inc., in Mountain View, CA. In the distant past he worked as a drummer in the Woody Herman Band and with the C. C. Riders. While a member of the Computer Systems Research Department at Bell Laboratories from 1977 to 1982, Dr. Chesson developed the packet driver protocol used by uu^{cp} and the mp^x files in the Seventh Edition of UNIX as well as the original protocols and software implementations for the Datakit network. At Silicon Graphics, he has implemented XNS and other network protocols and is currently developing new technologies. ■

Outstanding Learning Captivates, Motivates



And it saves you money.

USER TRAINING CORPORATION

CALL NOW (408) 370-9710

591 W. Hamilton Ave. • Campbell, CA 95008

UNIX* JOBS REGISTRY

National registry of candidates and jobs in the Unix field. Please give us a call; send a resume; or request a free Resume Workbook & Career Planner. We are a professional employment firm managed by graduate engineers.

800-231-5920

P. O. Box 19949, Dept. UR

Houston, TX 77224

713-496-6100



Scientific Placement, Inc.

*Unix is a trademark of Bell Labs

Circle No. 63 on Inquiry Card

Circle No. 26 on Inquiry Card

UNIX™ & 'C'

HANDS-ON TRAINING

SEMINARS • VIDEO-BASED TRAINING • AND NOW INTERACTIVE VIDEO



Select your training medium according to the type of training you prefer and the number of people to be trained.

VIDEO-BASED TRAINING. The Computer Technology Group's Video-Based Training integrates professionally developed and produced video and text material, as well as hands-on exercises, into complete training programs.

Our courses are produced with the highest standards of video quality, applying the latest techniques of instructional design including the use of computer graphics and animation techniques to compress learning time. The students' time is not wasted with the "camera at the back of the classroom" or "chalk talk" approach which is so inefficient, and often ineffective, in transferring skills.

Our Video-Based Training courses are completely self-contained, including the hardware-independent hands-on exercises. All you need is a video cassette player.

COURSE	Number of Modules	
	Video-Based	Interactive
Computers at Work	15	
UNIX—An Executive Perspective	1	
UNIX Overview	6	6
UNIX Fundamentals	15	15
'C' Language Programming	16	16



INTERACTIVE VIDEODISC TRAINING. Our new UNIX Videodisc Training Curriculum combines the benefits of our Video-Based Training with the flexibility of microcomputer access. Designed as a one-on-one tutor, our interactive system assesses the training needs of each student and dynamically tailors the training to his/her specific needs, thus eliminating redundant training. Through engaging exercises and interactive video, we are able to increase student comprehension while reducing training time.

Developed by the Computer Technology Group and Interactive Training Systems, the curriculum uses the latest laser videodisc technology—including IBM PC, color monitor and Interactive Training System Controller.

PUBLIC AND IN-HOUSE SEMINARS. Both public and in-house seminars are offered on a wide variety of UNIX and 'C' Language subjects, including:

UNIX Overview • UNIX Fundamentals for Non-Programmers • UNIX Fundamentals for Programmers • Shell as a Command Language • 'C' Language Programming • Shell Programming • Using Advanced UNIX Commands • UNIX Internals • UNIX Administration • Advanced 'C' Programming Workshop • Advanced 'C' Programming Under UNIX • Berkeley UNIX Fundamentals and "csh" Shell.

Call toll-free: (800) 323-UNIX
or in IL (312) 987-4082
310 S. Michigan Ave., Chicago, IL 60604

**COMPUTER
TECHNOLOGY
GROUP**

Telemedia, Inc.

TM UNIX is a Trademark of Bell Laboratories

INDUSTRY INSIDER

Five more roses

by Mark G. Sobell

The cover for the brochure for IBM's "Personal Computer Interactive Executive" (PC/IX) depicts a single rose standing in a vase. Add five more roses and you have the cover for the new IBM "Virtual Machine Interactive Executive" (VM/IX) brochure. Preliminary reports indicate that similarities between the two systems don't stop with the manual covers. After all, PC/IX and VM/IX are both versions of System III ported by Interactive Systems Corporation (ISC).

VM/IX runs on IBM's mainframe System/370 under the host operating system, VM/SP. This is IBM's fourth UNIX system in what appears to be a coordinated thrust into the UNIX market. Compatibility with PC/IX provides users with a migration path: a consistent operating environment from micro to mainframe. Although not as comprehensive, this path provides an alternative to the AT&T 3B line of machines. And it may grow more comprehensive yet: look to PC/XT 370, a PC capable of serving as a workstation for an IBM 370, to communicate directly with VM/IX in the not-too-distant future.



IBM cautions that VM/IX is a controlled release available only on a limited basis, but there can be little doubt of where it's headed. One of the limiting factors cited by IBM is general lack of technical experience with the product at local IBM branch offices. As IBM technical representatives learn more, VM/IX is likely to become more widely available.

Of necessity, IBM has packaged the System/370 with a Series/1 front-end processor to assist in communications. The System/370 can only accept synchronous, half-duplex, EBCDIC

communications, but the Series/1 interface allows VM/IX users to utilize asynchronous, full-duplex, ASCII lines.

One puzzling question that arises about this UNIX port, though, is: "Why did IBM go with System III and not System V UNIX?" There has been speculation that IBM is trying to establish System III as *the* standard in the face of AT&T's System V push. What's more likely is that ISC and IBM could not get their hands on a copy of System V when the port was started. If this is the case and sales of VM/IX go well, IBM will probably be announcing an upgrade to System V as soon as ISC is able to complete the port.

VM/IX itself is plain old (old?) System III without any icing. It contains no Berkeley enhancements (no **vi**, no **Cshell**, no **more**, no Berkeley **mail**). But, in addition to good old **ed**, ISC has provided **INed** as a system editor. The **lpr** printer spooler, meanwhile, has been replaced with a table-driven, general-purpose spooler named **print** that you can use to queue any output *or* input for any device.

An important area not addressed by the VM/IX release is

THE CODATA DIFFERENCE

A Major Improvement in OEM Microcomputers.

With so many microcomputers on the market, and with each manufacturer claiming to have the solutions to everyone's problems, why are more and more OEM users choosing CODATA?

Very simply because the CODATA 3300 works!

That's an important part of what we call THE CODATA DIFFERENCE - a refreshing, new concept in OEM marketing.

More performance...Lower price...Extended warranty.

We've increased the performance of the CODATA 3300 and, at the same time, we've lowered the price. That means our CODATA multi-user, multi-tasking 16-bit computer systems have price/performance ratios nobody else can match. We're so confident in our system's integrity that we've extended the warranty on every CODATA 3300 to 180 days. That's the longest warranty in the business and, let's face it, you've got to be pretty sure of your product to guarantee it for six full months. We're sure.

These are more elements of THE CODATA DIFFERENCE.

An uncomplicated pricing policy.

We have only two price sheets: one for suggested retail, the other for OEM prices. Simple, straightforward, easy-to-understand. Beyond that, there are no quantity commitments, no bill-backs, no hassles.

Full support and technical assistance if needed.

Our customers select the CODATA 3300 because it fulfills their needs, because it does what we say it will do, and because we have taken great pains to assure that it is functionally simple and capable of performing its assigned tasks with no problems. But, if technical support is required it's only as far away as your phone and available at a moments notice via our TOLL FREE number during normal business hours in every U.S. time zone.

A multi-user, 33-megabyte system for only \$9,600.

The CODATA 3300 is a powerful 16-bit, 68000-based, MULTIBUS system that can effectively accommodate up to ten users. It's a complete UNIX system that runs full ANSI standard FORTRAN-77, RM/COBOL, BASIC+, SMC BASIC, APL, and PASCAL. The 3300 provides up to 33 megabytes of unformatted on-line storage via an integrated, high speed Winchester drive, and a removable, quad-density 5-1/4" floppy disc system. The 3300 features 320 K bytes of parity protected RAM.

An expanded CODATA 3300, with 84 megabytes of disc storage is priced at \$13,500. The 3300 is also available as a 12-megabyte system for as low as \$7,800.

LET US SHOW YOU THE CODATA DIFFERENCE.

Write or call us for more information.



CONTEL CODATA

285 North Wolfe Road, Sunnyvale, CA 94086
1-800-521-6543. In California: 1-800-221-2265

KEYBOARD AND MONITOR OPTIONAL

penetration of academic environments. During the '50s and '60s, IBM was masterful at getting students used to working with its machines. As the students went into industry, their familiarity with IBM ultimately brought IBM into many data processing shops. If IBM is involved in UNIX for the long term and is making a serious attack on the UNIX market, it will need to provide a system that four-year schools can sink their teeth into (read as "Berkeley UNIX").

CONCLUSIONS

Chalk up another big one for UNIX. When Big Blue starts carrying AT&T software on its large machines, you can bet it is respon-

**Compatibility with
PC/IX provides
users with a
migration path: a
consistent
operating
environment from
micro to
mainframe.**

ding to pressure from some of its major customers. And IBM, at least in the past, has set the trend in operating systems (look at what the introduction of PC-DOS did to CP/M). IBM's announcement of VM/IX, on the heels of the PC/IX announcement, makes a strong case for a soaring UNIX market.

Mark G. Sobell in the author of "A Practical Guide to the UNIX System" (Benjamin/Cummings, 1984). His 10 years in the computer industry include programming and technical writing experience. Mr. Sobell has been involved in UNIX for four years and is currently a consultant in the San Francisco Bay Area. ■

**UNPARALLELED
PERFORMANCE
and PORTABILITY
in an ISAM PACKAGE
at an UNBEATABLE
PRICE**



c-tree™
BY FAIRCOM

2606 Johnson Drive
Columbia MO 65203

The company that introduced micros to B-Trees in 1979 and created ACCESS MANAGER™ for Digital Research, now redefines the market for high performance, B-Tree based file handlers. With c-tree™ you get:

- complete C source code written to K & R standards of portability
- high level, multi-key ISAM routines and low level B-Tree functions
- routines that work with single-user and network systems
- no royalties on application programs

\$395 COMPLETE

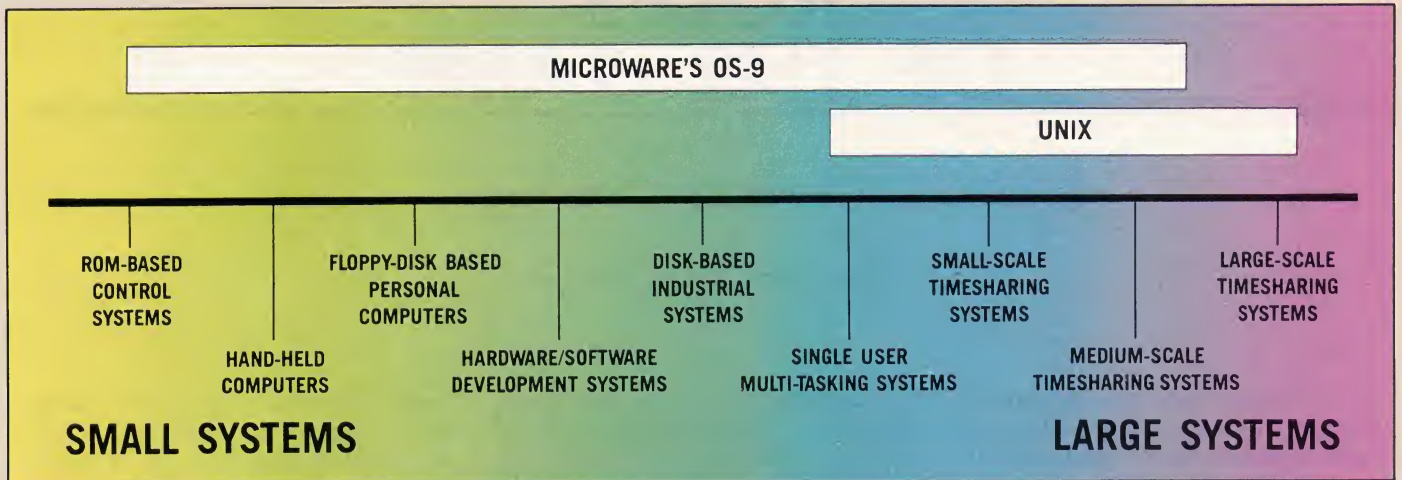
Specify format:
8" CP/M® 5¼" PC-DOS 8" RT-11

for VISA, MC or COD orders, call toll free
1-800-232-3344

Access Manager and CP/M are trademarks of Digital Research, Inc. c-tree is a trademark of FairCom.

© 1984 FairCom

Only Microware's OS-9 Operating System Covers the Entire 68000 Spectrum



Is complicated software and expensive hardware keeping you back from Unix? Look into OS-9, the operating system from Microware that gives 68000 systems a Unix-style environment with much less overhead and complexity.

OS-9 is versatile, inexpensive, and delivers outstanding performance on any size system. The OS-9 executive is much smaller and far more efficient than Unix because it's written in fast, compact assembly language, making it ideal for critical real-time applications. OS-9 can run on a broad range of 8 to 32 bit systems based on the 68000 or 6809 family MPUs from ROM-based industrial controllers up to large multiuser systems.

OS-9'S OUTSTANDING C COMPILER IS YOUR BRIDGE TO UNIX

Microware's C compiler technology is another OS-9 advantage. The compiler produces extremely fast, compact, and ROMable code. You can easily develop and port system or application software back and forth to standard Unix systems. Cross-compiler versions for

VAX and PDP-11 make coordinated Unix/OS-9 software development a pleasure.

SUPPORT FOR MODULAR SOFTWARE — AN OS-9 EXCLUSIVE

Comprehensive support for modular software puts OS-9 a generation ahead of other operating systems. It multiplies programmer productivity and memory efficiency. Application software can be built

from individually testable software modules including standard "library" modules. The modular structure lets you customize and reconfigure OS-9 for specific hardware easily and quickly.

A SYSTEM WITH A PROVEN TRACK RECORD

Once an underground classic, OS-9 is now a solid hit. Since 1980 OS-9 has been ported to over a hundred 6809 and 68000

systems under license to some of the biggest names in the business. OS-9 has been imbedded in numerous consumer, industrial, and OEM products, and is supported by many independent software suppliers.

Key OS-9 Features At A Glance

- Compact (16K) ROMable executive written in assembly language
- User "shell" and complete utility set written in C
- C-source code level compatibility with Unix
- Full Multitasking/multiuser capabilities
- Modular design - extremely easy to adapt, modify, or expand
- Unix-type tree structured file system
- Rugged "crash-proof" file structure with record locking
- Works well with floppy disk or ROM-based systems
- Uses hardware or software memory management
- High performance C, Pascal, Basic and Cobol compilers

microware[®]
OS-9[™]

MICROWARE SYSTEMS CORPORATION
1866 NW 114th Street
Des Moines, Iowa 50322
Phone 515-224-1929
Telex 910-520-2535

Microware Japan, Ltd
3-8-9 Baraki, Ichikawa City
Chiba 272-01, Japan
Phone 0473(28)4493
Telex 299-3122

OS-9 is a trademark of Microware and Motorola. Unix is a trademark of Bell Labs.

Circle No. 30 on Inquiry Card

RULES OF THE GAME

Seller beware

by Glenn Groenewold

Consider this scenario: you've created and marketed a program which has run beautifully on your computer and during extensive beta testing. However, an undetected bug produces calamitous effects if the program is run on the Goliath Z100x system. When one user does this, the ensuing crash results in the loss of inventory figures for a year-end sale, forcing cancellation of the event and the loss of thousands of dollars.

Or consider this: under a juicy contract, you have written the documentation for Goliath's system. Unfortunately there is an ambiguity in the wording of one of the key sections of your manual that a user misinterprets, resulting in a costly mistake. The user sues Goliath, naturally, and Goliath in turn sues you.

If this sounds far-fetched, consider that during the past 30 years or so, one of the major developments in American law has been the increasing ease with which consumers — even bystanders — can sue producers and providers of products and services. More and more, members of our society have shown a tendency to run to the courts to seek compensation for a whole range of grievances. And lest you think no one would bother to go to court over *your* product because of the small amount of



money involved, you're probably already aware that there's a type of lawsuit known as a class action in which a handful of product users can represent everyone who uses the product.

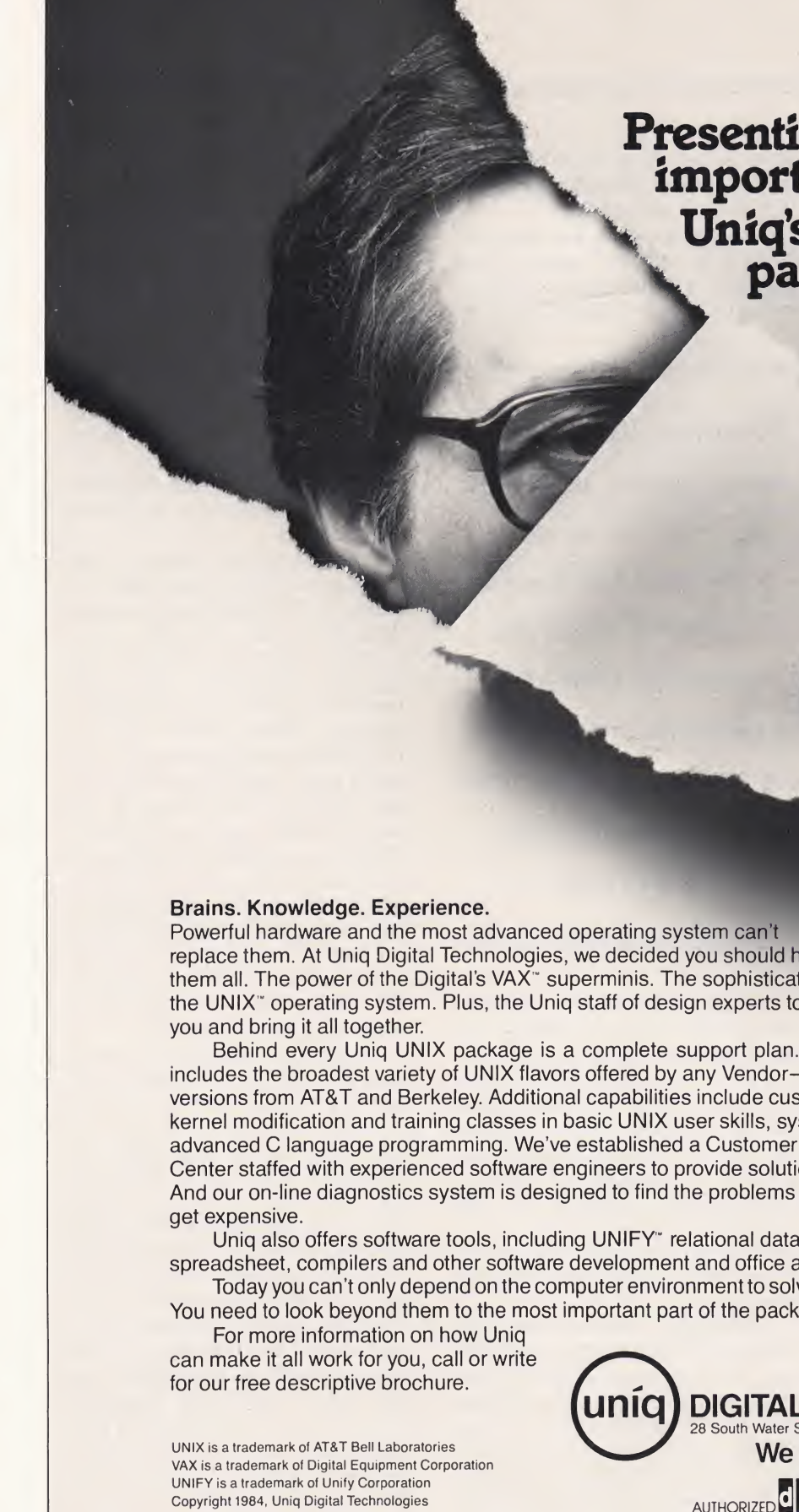
As recently as the early 1950s, it was still rather difficult for a consumer to sue successfully for a defective product. Formerly, there were a number of rules of law which in effect protected producers and providers of goods and services. From the standpoint of contract law, there was a rule which said that only an immediate party to a contract could sue under it. This *privity* requirement often made it impossible to sue a manufacturer, since most sales transactions took place between the consumer and a dealer without directly involving the

manufacturer. Negligence suits were also not generally regarded as a profitable means of proceeding against a manufacturer because a number of legal requirements too complicated to discuss here served to insulate the manufacturer from this type of liability.

But things have changed. Barriers to successful consumer lawsuits have been dropping right and left. For one, the privity requirement has been eroded to the point where there's not much left. This has opened the way for consumer suits on a contract basis. For another, the requirements for negligence suits have been liberalized. And there is still another basis for consumer lawsuits, products liability, which originated in the idea of imposing legal liability for inherently hazardous products or situations — such as the manufacture of explosives or the keeping of dangerous wild animals. Over the years, products liability has been expanded to the point where it can even apply to defective computer programs.

LIABILITIES

Liability of one party to pay damages to another ordinarily arises under one of two distinct areas of law. First, there is liability arising out of a contract. Though this requires that there *be*



**Presenting the most
important part of
Uniq's UNIX™
package.**

Brains. Knowledge. Experience.

Powerful hardware and the most advanced operating system can't replace them. At Uniq Digital Technologies, we decided you should have them all. The power of the Digital's VAX™ superminis. The sophistication of the UNIX™ operating system. Plus, the Uniq staff of design experts to work with you and bring it all together.

Behind every Uniq UNIX package is a complete support plan. Support that includes the broadest variety of UNIX flavors offered by any Vendor—including the latest versions from AT&T and Berkeley. Additional capabilities include custom driver development, kernel modification and training classes in basic UNIX user skills, system administration and advanced C language programming. We've established a Customer Technical Assistance Center staffed with experienced software engineers to provide solutions to those critical calls. And our on-line diagnostics system is designed to find the problems before the solutions get expensive.

Uniq also offers software tools, including UNIFY™ relational data base, Unicalq electronic spreadsheet, compilers and other software development and office automation products.

Today you can't only depend on the computer environment to solve your company's problems. You need to look beyond them to the most important part of the package. Support. Uniq support.

For more information on how Uniq can make it all work for you, call or write for our free descriptive brochure.

UNIX is a trademark of AT&T Bell Laboratories
VAX is a trademark of Digital Equipment Corporation
UNIFY is a trademark of Unify Corporation
Copyright 1984, Uniq Digital Technologies



DIGITAL TECHNOLOGIES

28 South Water Street, Batavia, IL 60510 • (312) 879-1008

We make it work.

AUTHORIZED **digital**® COMPUTER DISTRIBUTOR

a contract, these are plentiful enough. Every time we purchase something, we are automatically party to a contract of sale which can give rise to certain rights and obligations.

The other broad area of law under which legal liability commonly comes into existence is called torts. Though the term often strikes non-lawyers as faintly amusing, there's nothing funny about being on the losing end of one of these lawsuits. The point to remember is that tort liability has nothing to do with contracts; it springs from the obligation we have to refrain from harming others.

These days, the tort actions we most frequently hear about are lawsuits based on negligence — personal injury suits being prime examples. But there is another branch of tort law which also concerns us, that of products liability. While at one time this was an exceedingly restricted basis for lawsuits, it's been expanded dramatically in recent years. Products liability can be a potent weapon for the dissatisfied consumer, because it doesn't involve proving that the manufacturer was negligent — only that the product was defective.

PROTECTING YOURSELF

What all this means is that someone who provides a product or service these days is going to have to deal with the prospect of being sued by somebody whose very existence is unsuspected. And even an unsuccessful suit can be a major pain for the defendant.

So what can you do?

First of all, you must figure out what sort of legal hazard you may be exposed to. This will depend upon the type of activity you're engaged in. For instance, if you are *selling* a product, you will

have to be concerned about contract liability. Most importantly, this means warranties.

At this point, it becomes difficult to be specific, because contract law is primarily a state matter, providing us with 50 different

Products liability can be a potent weapon for the dissatisfied consumer because it doesn't involve proving that the manufacturer was negligent — only that the product was defective.

versions (not to mention assorted territories). Though almost all states and territories have adopted a set of laws known as the Uniform Commercial Code, many have made their own changes so that the laws are no longer really uniform. The important thing to remember is that if you do business across state lines, you'll be held to the requirements of every state in which you sell your product.

Generally, you must be concerned with two kinds of warranties. *Express* warranties are easy to understand. If you represent that your program will do certain things on the Goliath Z100x, you can be sued for breach of warranty if it doesn't. *Implied* warranties

can be trickier. There are several of these, but the most important is the warranty of *fitness for a particular purpose*. Let's say, for instance, that you're selling software under a trade name such as Goliath Sooper-Speller through retail outlets primarily devoted to Goliath products. It certainly would be reasonable for a buyer to assume that your software was designed for use with the Goliath computer. If in truth your program causes the Goliath system to crash, you could have a date in court.

But suppose instead that you're *licensing* your software, not selling it. Now the law gets more complex. As regards your licensee, you have various contractual obligations, and your licensee can sue you for breach of contract. Other people who are ultimately damaged through the use of your software would probably have to sue you on a tort basis, though, either on a product liability or a negligence theory. In some states, they may be able to collect damages for such things as lost potential profits, but this is not true in every state. The warranties of the Uniform Commercial Code have no application under a license, however.

Once you've figured out what kind of liability you might have, you can decide what to do to try to protect yourself. If you're a seller, you can attempt to get out from under any warranties by using language such as, "Seller makes no warranty of any kind, express or implied, concerning the use of this product." This may or may not work, depending on the situation and the state law you're faced with, but it probably can't do any harm. (If you figure that you have significant potential liability under a warranty, it'd be a good idea to have a chat with a lawyer.) You can also try to limit your liability under contracts,

The Firebreathers from Gould blast the competition into oblivion.

With blazing performance.

Great leaps in raw performance are rare in the computer world. Usually, changes occur in increments of half-a-MIP or so. Now real Firebreathers roar into the arena, quadrupling the best the competition has to offer.

With scorching speed.

These creatures don't run a little faster. Running real production code supplied by VAX™ users, our PowerNode™ 9000 benchmarks at 4.5 times faster than the VAX 11/780...at a comparable price! Even our second-in-line

PowerNode 6000 runs 1.5 times faster. But with all this power and speed, the PN6000 has a dainty footprint...60% less than the VAX 11/780.

With sizzling simplicity.

Even the hottest firebreather won't fly if it's hard to run. Ours are UNIX*-based, so ease of use and compatibility become part of the power. Plus we offer the "Compatibility Suite", application software packages which are consistent across our entire Gould PowerSeries™ product line...the widest range of UNIX-based systems in the world.

More than a myth.

We know these claims sound outrageous. Make us prove them. Give us your benchmarks and production code. We'll show you

numbers that surpass all expectations. Call 305-797-5459 for a test flight on the Firebreathers. Mainframe performance disguised as a supermini. The scorching performance you expect from Gould.

Gould Inc.,

Computer Systems Division
Distributed Systems Operation
6901 West Sunrise Boulevard
Ft. Lauderdale, FL 33313

*UNIX is a trademark of AT&T Bell Laboratories

™PowerNode and PowerSeries are trademarks of Gould Inc.

™VAX is a trademark of Digital Equipment Corp.



GOULD
Electronics

XENIX Communications Available NOW!

Put your computers on speaking terms.



Introducing **TERM.** Communications Software

Everyone from the beginning computer user to the expert finds TERM easy to learn and powerful to use. Just plug it in and go! In a few keystrokes you can access a remote database or send a group of files to another system.

TERM allows your computer to perform efficient, error-free exchange of binary or text files, over phone lines or hard-wired circuits at speeds of up to 9600 baud. Available options allow you to include or exclude a group of files for transfer in a single command.

TERM's "data capture" feature allows saving transcripts of sessions with remote mainframe and minicomputers to disk for later editing or printout, if desired.

- Pre-installed and ready to run
- Automatic error checking and re-transmission
- Wildcard (* *) file send/receive capability
- Xon/Xoff, Etx/Ack, Ascii protocols for communications with non-TERM systems
- Full/half duplex emulation mode for remote systems
- Modem7 protocol for remote bulletin boards
- Auto-dial/Answer and Hangup supported on Hayes Smartmodem 300/1200 and compatibles
- Programmable batch file capability
- Unattended file transfer/auto logon
- Translation tables for input and output

TERM is available now on the Altos 586 and Tandy Model 16, along with more than 35 CP/M—80, MSDOS, and CP/M—86 systems; IBM PC*/XT, Kaypro, Osborne, Televideo, Victor, Apple* II—CP/M, Heath, Vector, Sanyo, Eagle, Molecular, Altos, and many others.



CALL OR WRITE FOR FREE PRODUCT CATALOG
CENTURY 9558 South Pinedale Circle
S O F T W A R E Sandy, Utah 84092
(801) 943-8386
We make it easy for you. Circle No. 33 on Inquiry Card

CP/M is a registered TM of Digital Research

Computer Scientists

The Lawrence Livermore National Laboratory is a research and development facility operated by the University of California for the U.S. Department of Energy. Located in the San Francisco East Bay Area, the Laboratory employs 8,000 individuals engaged in challenging basic and applied R&D projects.

The Laboratory has current openings for application and systems programmers with Unix experience to work in the following areas:

NETWORKING: Be part of the effort to develop a network connecting distributed resources throughout the Laboratory to obtain efficient computer-to-computer communications. One such major resource is the LLNL Octopus Network which currently contains eight of the latest generation supercomputers, including the Cray-XMP.

SCIENTIFIC WORKSTATION DEVELOPMENT: On this program you will be part of a team to develop an intelligent terminal to be used for interactive work and communication with other workstations, computer centers and networks.

The professionals we are seeking for these challenging assignments should have a BS degree in computer science, electrical engineering, or a related field. Requires at least one year Unix and "C" programming experience and the ability to work effectively with other group members. For Systems Programmers, an interest in all aspects of systems design, development, implementation and maintenance is essential.

To apply for these positions, qualified applicants should direct their resumes to: **Sue Porter, Professional Employment, Lawrence Livermore National Laboratory, P.O. Box 5510, Dept. KUR-914, Livermore, CA 94550.**

U.S. Citizenship is required.
An equal opportunity employer m/f/h/v.

University of California
**Lawrence Livermore
National Laboratory**

such as licensing agreements, by including a clause stating that the other party is to hold you harmless for any damages.

There are two routes you can take to try to protect yourself from tort liability: first, you can do your best to avoid marketing a defective product or being negligent in

Barriers to successful consumer lawsuits have been dropping right and left.

providing a product or service. Second, you can consider insurance. Obtaining useful coverage can be a problem in a burgeoning new field where the risks are not clearly understood. Even when available, policies may be prohibitively expensive or may be so riddled with exclusions as to be practically worthless. If you elect to go for insurance, search out a knowledgeable broker whom you feel you can trust, and ask lots of questions.

As if all this hasn't been sufficient to guarantee insomnia, there are other legal hazards lurking in businessland that we haven't discussed. You'll have to keep a wary eye on anti-trust laws for instance, if you intend to do exclusive licensing. Franchising, meanwhile, has a whole list of perils all its own. But these and other horrors will have to wait for future consideration.

Glenn Groenewold is a California attorney who devotes his time to computer law. He has served as an administrative law judge, has been active in trial and appellate work and has argued cases before the state Supreme Court. ■

the Cray style...

Starts With You.

Your knowledge and skills put to their best use. Your willingness to experiment with new ideas. Your ability to meet the challenges put forward by the company synonymous with the term "Supercomputer." That's your style. And that's the Cray Style.

Cray Research is the world leader in the design, manufacture and marketing of supercomputers. We are a multi-national company meeting high-speed computing needs in a variety of research areas. Cray computers are allowing breakthroughs in such fields as aerodynamics, automotive design, geological research, weather forecasting and high speed graphics.

Software Engineer

Due to expansion and an increased emphasis, there currently exists within our Software Development Division a unique opportunity to join our team working to develop UNIX to run efficiently on large mainframes.

Principle responsibilities will be to adapt UNIX to our supercomputer environment especially in the area of I/O performance, networking to other mainframes, support for multiprocessors, and multi-tasking. Consideration for this unique ground breaking opportunity will be given to you if you possess 7 or more years of experience in software development including 3 or more years development of UNIX at the kernel level. Communications (networking), UNIX multi-processing, and large mainframe experience will be real plusses.

Technical Support Analyst

UNIX has been selected as the operating system for our most advanced supercomputer. That development has created within our Software Technical Support Division a unique ground floor opportunity. We now have a need for an individual interested in applying support skills and UNIX internals experience to state-of-the-art computing.

In this challenging role, you will ultimately assume responsibility for all UNIX internals inquiries escalated to headquarters support. Early installations are limited in number.

At least two years UNIX internals experience, willingness to travel (up to 25 percent) and good communication skills will qualify you for consideration.

Cray Research offers an excellent compensation package and salary commensurate with experience. For confidential consideration, submit a resume or letter outlining qualifications to our Minneapolis-St. Paul location.

Marc Scott
Department 901-U
CRAY RESEARCH INC.
1440 Northland Drive
Mendota Heights, MN 55120
Equal Opportunity Employer M/F/H/V

CRAY
RESEARCH, INC.

/usr/lib

What might have been

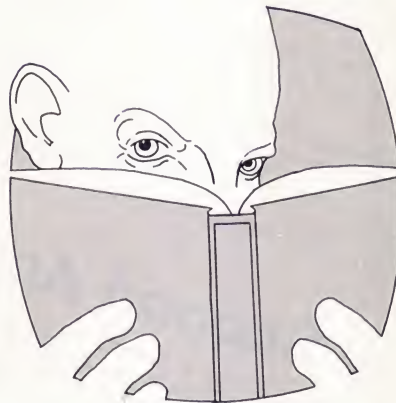
by Jim Joyce

UNIX — The Book

UNIX -- The Book (Sigma Technical Press, 1982[!], 265pp) is what *should* have been reviewed in the April/May issue of *UNIX REVIEW* instead of *The UNIX Operating System Book*. Both are attributed to the same authors, Mike Banahan and Andy Rutter, and presumably describe the same book. But the sad, sad story is that, in order to gain US distribution, Banahan and Rutter turned their book over to John Wiley & Sons, Inc., where it was turned into a mish-mash of error and misunderstanding.

I found *UNIX -- The Book* to be a funny, friendly, *accurate* and quite useful introduction to UNIX. Nervous editors at Wiley may have made changes because they felt the authors' breezy humor may not be to everyone's taste. But their humor *is* in good taste, and it manages to illustrate serious points with a clever turn of phrase, as shown in this sage wisdom about picking an experienced UNIX person to ask for advice:

Pick your guru carefully, like a horse. Go for one with an alert expression, bright eyes and a quiet voice: if they type with ten fingers, it's a good sign. Avoid at all costs the terminal freak; bags under



the eyes, a glazed expression and an unwashed appearance are signs to beware of. This breed thinks it knows everything and will tell you so interminably. Even if they [sic] do know it all, they generally can't explain anything without reference to even more complicated examples that you won't understand either.
(p. 1)

Though an overgeneralization about appearance, this passage contains sound advice about picking a consultant! The authors also make themselves the topic of gentle fun:

Occasionally you may come across the UNIX 'expert'. Too old and weary for anything of use, these

creatures are put out to grass, where they immediately elect themselves to committees and start writing books. If you have one of these, be kind to it, buy it drinks and treat it with respect. On no account ask it for advice, for its language is no longer the same as yours.
(p. 2)

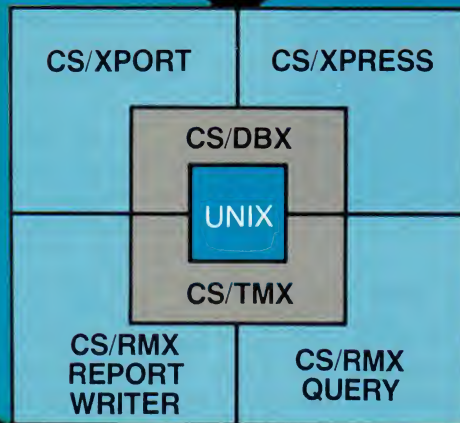
But *seriously*, you might ask, can one *really* learn UNIX from a book with such an irreverent tone? The answer is: Yes, please!

This is one of the few books that makes the important point that the shell is just another program rather than a fixed part of the operating system. And, to make sure the point isn't missed, the authors include the following in a group of questions at the end of the same chapter:

5) *What is the shell? How does it differ from a program that you might have written yourself?*

How refreshing! Alas, points such as this are sadly lacking in the book released for consumption in the United States. *The UNIX Operating System Book*, has little in common with the original book, apart from the names of Banahan and Rutter on the cover. Few of their insights are left intact. No doubt the publisher had good intentions, but the review of the

Extend the power and performance of UNIX™ with CS/XTEND



Cincom Systems, the industry leader in data base/data communication software systems, passes the power of its proven high performance software technologies for mainframe and minicomputers to the world of UNIX. CS/XTEND, our new fully integrated family of development and decision support components, makes it easy to create and implement even the most complex applications. And, our experience and expertise in developing quality software ensure that you get the high performance you need while using a minimum of machine resources.

CS/XTEND includes integrated solutions for professional system developers as well as productivity tools for non-technical information users:

CS/DBX—a high performance shared Data Base Executive with the power to handle multiple users concurrently accessing any number of files.

CS/XPORT—a Distributed System Interface that links computers so they can share data, providing the foundation for distributed data base networking.

UNIX is a trademark of AT&T Bell Laboratories

CS/TMX—a powerful Terminal Management Executive that simplifies interactive programming by making programs independent of the terminal, and provides overall network management.

CS/RMX—an easy-to-use Retrievals Management Executive that provides relational access to the data base for interactive inquiries and report writing by both end-users and computer professionals.

CS/XPRESS—a dynamic Application Builder that lets non-programmers automatically generate, maintain and access custom-built files of information.

CS/XTEND is fully compatible with Cincom's TOTAL® Data Base Management System. That means you can implement applications to operate on different types of computers using identical data base techniques.

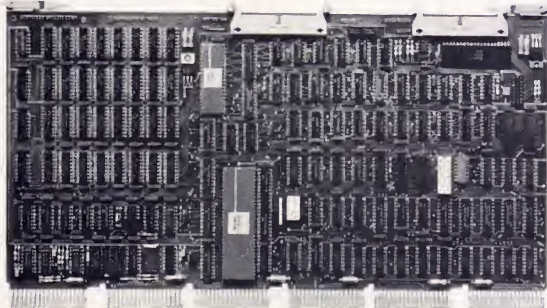
Find out today how CS/XTEND extends the power and performance of your UNIX system. Simply call or write, Cincom Ventures Division, 2300 Montana Avenue, Cincinnati, Ohio 45211.

800-543-3010 (In Ohio: 513-661-6000)



Cincom Systems

DEC → TO IBM/SNA



Full SNA capability for your DEC computer! Comboard™/SNA gives your terminals access to IBM interactive applications. Data can be transferred between systems, all in the complete *fully supported package*. Comboard/SNA from Software Results.

Proven and reliable, Comboard/SNA is a single-board 256kb communications computer that plugs into your DEC Unibus. Teamed with Comboard software the system is a *cost-effective* solution to troublesome SNA communications problems.

Your DEC emulates an IBM PU Type 2 communication node. You have a *full gateway into your SNA* without passing through a secondary network.

For further information call or write Software Results... the leader in DEC to IBM communications.

COMBOARD™

Communications Results from
**SOFTWARE
RESULTS
CORPORATION**

Call Toll-free
1-800-SRC-DATA


(1-800-772-3282)
In Ohio call collect, 1-614-267-2203

2887 Silver Drive Columbus, Ohio 43211 Telex: 467-495 SRC DATA CI

COMBOARD is a Trademark of Software Results Corporation

DEC UNIBUS is a Trademark of Digital Equipment Corp.

Circle No. 35 on Inquiry Card

 /usr/lib

Americanized edition offered in my April/May column reveals what can happen when authors allow their material to be tampered with and then do not review the result.

But back to the *real* Banahan and Rutter book. Since this issue of *UNIX REVIEW* centers on system administration, we turn to the chapter on "Maintenance" to see how they approached the topic.

Because the book is aimed at being introductory, the chapter is not intended to be a full system administration manual, but an explanation of the ideas a good system administrator should know. The explanation of how the system starts up and shuts down lends important insight into what goes on in these two important activities, rather than dwelling on details about the procedure.

Details system administrators will want to know are included in Chapter 8, "The Process Environment," and Chapter 9, "Libraries."

Published by Sigma Technical Press, in Cheshire, and distributed by John Wiley & Sons, Ltd., Sussex, this book should be distributed in the US — and all copies of the other version should be withdrawn. Mike Banahan, in a telephone call from London, said he and Rutter erred in refusing to have anything to do with the US edition of the book once they were informed (!) changes would be made to it. He said they now know the error of their ways. I hope John Wiley & Sons learns soon, too.

The Table of Contents for *UNIX — The Book*

1. In the Beginning (12pp)
 2. Files and Simple Commands (30pp)
 3. The Editor, Ed (17pp)
 4. C (28pp)
 5. The UNIX Filestore (12pp)
 6. Software Tools (13pp)
 7. Text Preparation (20pp)
 8. The Process Environment (17pp)
 9. Libraries (17pp)
 10. Maintenance (19pp)
- Appendices
- A. General Commands (20pp)
 - B. The Editor (6pp)
 - C. Shell Syntax (11pp)
 - D. Standard Libraries (25pp)
 - E. System Calls (8pp)
 - F. The ovp program (6pp)
- Answers to questions (2pp)
References (1p)
Index (2 pp)

REAL WORLD UNIX

The trouble with *Real World UNIX* (Sybex, 1984, 209pp, \$16.95) is that author John D. Halamka ap-

parently doesn't know regular UNIX, let alone "real world UNIX." In contrast to Banahan and Rutter, Halamka confuses the relationship of the shell to UNIX:

UNIX comprises two basic elements, the shell and the kernel, the "outside" and the "inside" of the system. (p. 11)

Not stating that the shell is a program that can be replaced by other programs might seem a polite fiction intended to allow the authors to get on with introductory remarks, but it can cause readers to misunderstand a major source of UNIX flexibility — the fact that the shell can be any program, even `/usr/games/rogue`.

Real World UNIX, could have used a technical writer — or a freshman composition grader — to clean up its act a bit. The statement "UNIX and Ethernet have been developed for each other..." sounds like a ludicrous technical error until it finally dawns that the writer really meant the two are

compatible, complementary, and that Ethernet can link UNIX systems. Gee, whiz.

It is harder to attribute as bad writing the statement, "UNIX provides 'record locking'..." Thanks to John Bass there is a `/usr/group` standard for record locking, but vanilla UNIX has no such thing.

AT&T Technologies and the University of California at Berkeley will no doubt be surprised to learn that "System V [has] an extended shell developed at the University of California at Berkeley" (p. 12). It is well-known that many of the System V.2 utilities are warmed-over rewrites of Berkeley utilities, but the System V shell *is* the Bourne shell.

I regard a book as dangerous when, in addition to typographical errors (a relatively minor sin) and technical errors (a much greater sin), there are errors of judgement. Because UNIX is a flexible computing environment, users who use the full system need to have a good idea of what are and are not good practices. The applications example Halamka includes on page 21 is a prime example of showing a

U N I X S O F T W A R E

**UniPress
Product
UPDATE**

**LATTICE® C NATIVE AND CROSS
COMPILERS FOR THE 8086**

Outstanding software development tools

**Lattice C Cross Compiler
to the IBM-PC**

- Highly regarded compiler producing fastest and tightest code for the 8086 family.
- Use your VAX or other UNIX machine to create standard Intel object code for your 8086 (IBM-PC)
- Full C language and standard library, compatible with Unix.
- Small, medium, compact and large address models available.
- Includes compiler, linker, librarian and disassembler.
- 8087 floating point support.
- MS-DOS 2.0 libraries included.
- Send and Receive communication package optionally available.

Hosted On

Prices: VAX/Unix and VMS	\$5000
MC68000/8086	3000
Send and Receive	500

**Lattice C Native Compiler
for the 8086**

- Runs on the IBM-PC under MS-DOS 1.0 or 2.0.
 - Produces highly optimized code
 - Small, medium, compact and large address models available.
 - Compiler is running on thousands of 8086 systems.
- Price: \$425

COMING SOON

Amsterdam Compiler Kit

- Package of compilers, cross compilers and assemblers.
 - Full C and pascal language.
 - Generates code for VAX, PDP-11, MC68000, 8086 and NSC16000.
 - Hosted on many Unix machines.
 - Extensive optimization.
- Price: Full system \$9950
Educational Institution 995

Plink—For Lattice Native

- Full function linkage editor including overlay support.
- Price: \$395

OEM terms available • Much more Unix software, too! • Call or write for more information.

UniPress Software, Inc.

2025 Lincoln Highway, Edison, NJ 08817
201-985-8000 • Order Desk: 800-222-0550
(outside NJ) • Telex 709418
Mastercard and Visa

Lattice is a registered trademark of Lattice, Inc.
Unix is a trademark of Bell Laboratories. MS-DOS is a trademark of Microsoft.

See us at UNIX SYSTEMS EXPO/84, Los Angeles, Booth #544

Circle No. 36 on Inquiry Card

U N I X S O F T W A R E

Tektronix And Oregon — A Winning Combination

- Outstanding Careers
- Affordable Housing
- Clean Air

We have it all. Join us. TEKTRONIX is listed among the 100 best companies to work for in America and we have some outstanding career opportunities available for Software Engineers.

Our experienced Software Engineers are involved in generating new codes, making improvements and making bug fixes to UNIX™ kernel, device drivers, commands, utilities, and network programs. Our primary focus is on the 4.2 BSD enhancements.

The individuals we're seeking must possess leadership potential in a UNIX™ environment, be highly skilled as a C programmer, have in-depth knowledge of UNIX™ internals, VAX and PDP-11 architectures, along with related peripherals. An advanced degree is preferable, but the equivalent in related experience is acceptable.


TEKTRONIX is a Fortune 500 company that provides its professionals with competitive salaries, generous benefits that include profit sharing, health and dental insurance, and liberal educational assistance. Oregon offers natural unspoiled beauty, affordable housing, and unsurpassed outdoor recreation. For immediate consideration, please send your resume to Michele Goza, M.S. 46-943, Tektronix, Inc., P.O. Box 500, AMK1, Beaverton, OR 97077.

We are an equal opportunity employer m/f/h.

UNIX is a TM of Bell Laboratories.

Tektronix
COMMITTED TO EXCELLENCE

Circle No. 37 on Inquiry Card

 /usr/lib

bad practice. It shows the root directory, and sprouting from it are two directories, **payables** and **receivables**.

Even being charitable and believing that each of these directories is on its own file system, this business of cluttering up the root directory with sensitive names seems to be asking for someone who is a security risk to attempt something evil. On the other hand, if sensitive directories such as **payables** and **receivables** were in fact on the root file system, the author is asking for trouble. Consider, for example, a system failure, in which a damaged root file

**Halamka is not using
the file system with the same
sensitivity with which
it was designed.**

system superblock might mean the difference between not having **payables** or not having **receivables** — or both. Halamka is not using the file system with the same sensitivity with which it was designed.

The bad choice of directory allocation is further compounded on page 24 where we see, as one of the earliest commands introduced:

```
rm /payables/monolith_ind
```

It seems unlikely anyone would wish to remove a **payables** file without an awfully good reason. None is given. The lack of good judgement in choosing a more realistic example makes a mockery of the book's very title.

There is a chapter on system administration as long as the one in Banahan and Rutter, but there all similarity ends. Typographic errors (at least I *hope* they were typographic) abound in the examples of **/etc/passwd** entries. The colon-separated fields have blanks after the colons — no doubt added for readability — but not for technical accuracy.

When I first saw the example:

```
ceo: sdfsd34: 1: 1: John Halamka: /usr/ceo
```

it seemed unlikely that the encryption algorithm would stand for a blank in the field, and when I experimented, it did not work.

Halamka also includes the pronouncement that

userids must be a unique number "from 1 to 255." The system I am writing this review on has **userid**s as high as 340, and another system I use has **userid**s in the 900s. These are not simply integers the system uses — they can also be helpful organizational numbers for system administration use. However, don't go wild using **userid** numbers since the size of `/usr/adm/wtmp` reflects the magnitude of the largest **userid**.

One last nit before I leave off with this book. On page 103, the author states:

C, the language in which UNIX itself is written, is very complex and difficult to learn.

But, on page 163, in the Appendix on "UNIX Resources," we do not find an entry for Kernighan and Ritchie's C book nor anyone else's — even Kochan's quite readable book. I think the problem is that the author himself does not know C.

I generally put notations about the qualities of books on the title pages to remind me of where to find things. For this book, the page was practically obliterated with inked notations detailing errors.

For information on "real world" UNIX try books by Banahan and Rutter, Bourne, McGilton and Morgan, Sobell — but not this book.

The Table of Contents for *Real World UNIX*

Introduction (9pp)

1. Introduction to Hardware and Software Concepts (8pp)
2. UNIX Concepts (18pp)
3. Using the UNIX Shell (35pp)
4. System Administration (20pp)
5. Shell Programming (15pp)
6. Real World Hardware and Software (29pp)
7. The Future of UNIX (11pp)

Appendices

- A. UNIX Resources (18pp)
- B. One-Minute UNIX (21pp)
- C. Glossary (15pp)

Index (5pp)

A BOOK ON C

Al Kelley and Ira Pohl have put together a book on C (Benjamin Cummings, 1984, 362pp, \$21.95)

U N I X S O F T W A R E

**UniPress
Product
UPDATE**

EMACS™ & MINIMACS™

Powerful Text Editing Software for MS-DOS™, UNIX™ and VMS™

EMACS

- Acclaimed Gosling Version
- Runs under UNIX, VMS, and MS-DOS
- Full screen multi-window editor allowing several files to be edited simultaneously.
- Extensible and customizable via macros and built-in MLISP programming language.
- C, Pascal, and MLISP programming assist.
- Communicates with UNIX and VMS systems. Allows shell/DCL windows, and command execution with output directed to a window. For example, Emacs permits program compilation with source code in one window and compiler errors in another for ease of debugging.
- EDT emulation on VMS, optional on UNIX.
- Available on a wide range of hardware configurations including the VAX, 4.1/4.2, System III/V, Sun, Pyramid, Plexus, Cyb, Callan, Masscomp, Apple Lisa, Tandy Model 16, Pixel, Dual, Apollo, Integrated Solutions, Perkin Elmer, HP9000, NCR Tower, Cadmus, Momentum, Charles River Data, Fortune and more.

Prices: UNIX: \$395/binary • \$995/source
VMS: \$2500/binary • \$7000/source
MS-DOS: \$375/binary • \$995/source

See us at UNIX SYSTEMS EXPO/84, Los Angeles, Booth #544

MINIMACS

- Faster and smaller model Gosling Emacs available for Unix.
- Full screen editing with multiple windows.
- Full EMACS macro capability and keyboardings. Communicates with Unix via command execution with output directed to a window.
- Minimacs brings the powerful capabilities of the finest screen editor to smaller machines.
- Minimacs minimizes your resource load and is much smaller than the vi editor.

Price: \$375/binary • \$795/source.

OEM terms available. • Much more Unix software, too! Call or write for more information.

UniPress Software, Inc.

2025 Lincoln Highway, Edison, NJ 08817
201-985-8000 • Order Desk: 800-222-0550 (outside NJ)
Telex 709418 • Mastercard and Visa

Emacs and Minimacs are trademarks of UniPress Software, Inc. Unix is a trademark of Bell Laboratories. VMS is a trademark of Digital Equipment Corp. MS-DOS is a trademark of Microsoft.

Circle No. 38 on Inquiry Card

U N I X S O F T W A R E

that is basically quite good. They start off in a rather troublesome way, though, with a system dependent program! The results, as captured by Mark Horton's script program follow:

```
% cc trad.c -o trad ; trad
llehw ,odlro%
```

Pretty wonderful, eh? It is supposed to say:
hello, world

The “%” jammed up against the “odlro” is the prompt, since there was no newline character anywhere in the control string — a bit of carelessness, perhaps. I thought I was in for yet another bad book on C but I'm happy to report that the authors later turned me into a believer.

Their technique of presenting a program followed by a step-by-step dissection of each significant point is quite good. Their programming style is also good, but some of their examples are almost as questionable as the first one discussed here.

Also, I strongly disagree with their first example of recursion:

```
main()
{
    printf("The universe is never ending!");
    main();
}
```

This is infinite regression that continues until the RUB key is mercifully pressed, but it is not a true recursion as there is no test condition that stops the regression. That the authors encourage the reader to “try the program” without indicating how to stop it once it is running is strange indeed, and it is disturbingly reminiscent of the first example program cited earlier.

But then again, Kelley and Pohl are charming in the preprocessor section of the same chapter where they present the following “syntactic sugar”:

```
#define EQ ==
```

as a suggestion to guard against the potential confusion of logical equivalence == and assignment =.

ONE DAY CAN SAVE YOU YEARS OF SOFTWARE DEVELOPMENT EXPENSE

Learn how object-oriented programming and Objective-C,™ an evolutionary enhancement of C Language, will dramatically increase the productivity of your programming staff by making software reusability a practical reality.

OVERVIEW FOR EXECUTIVES

*Stamford, CT
October 2, 1984*

*Washington, DC
October 29, 1984*

All text materials and lunch are included in the \$200 tuition fee. Enrollment is limited. Call for registration, information and additional course listings.

Productivity Products International

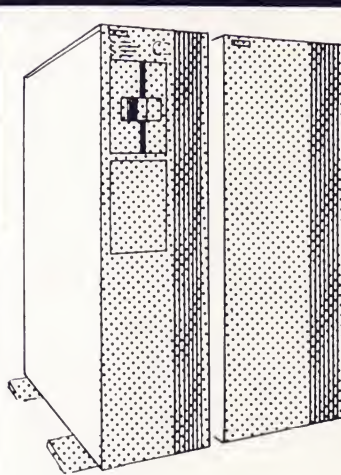


27 Glen Road
Sandy Hook, CT 06482
(203) 426-1875

Circle No. 38 on Inquiry Card

TOWER™ POWER

Give your Tower more Power!



- 80MB and 300MB cartridge disk subsystems
- 160MB and 300MB fixed media disk subsystems
- DIBOLIX™ — DIBOL™ for UNIX™
- UNIFY™
- Personal Secretary™ Word Processor

Contact: **SHA Computers, Inc.**

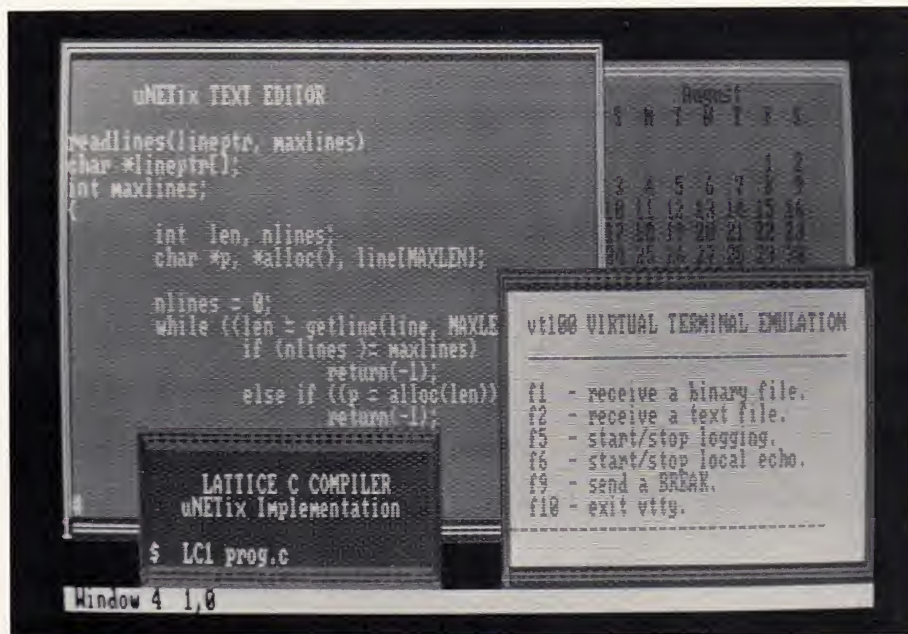
RD#3, Box 51, Tait Road
Saratoga Springs, NY 12866
(518) 587-5886

Trademarks: Tower, NCR Corporation; DIBOLIX, C/DIBOL Ventures; UNIX, AT&T; DIBOL, Digital Equipment Corporation; UNIFY, Unify Corporation; Personal Secretary, Finished Software.

Circle No. 39 on Inquiry Card

SOFTWARE DEVELOPERS

LESS THAN \$400 WILL TURN YOUR PC INTO A POWERFUL DEVELOPMENT WORKSTATION.



Until now, software developers who wanted the power of high-priced development systems had to pay a high price for it. There was no choice.

Now, there's a solution... whether you're developing programs to run on minicomputers or micros, and regardless of what target operating system you're using.

If an IBM® PC, PC XT™, or compatible is part of your development environment, the uNETix Software Development ToolKit will increase your productivity.

LANTECH SYSTEMS has put an end to the "no choice" problem with a collection of powerful software products which make your PC a versatile, but low cost development workstation.

The uNETix Software Development ToolKit contains:

- **uNETix SFS** — the powerful multi-tasking, UNIX™ compatible operating system, which is at the heart of the LANTECH SYSTEMS product line. The unique "Window Management" feature provided with uNETix allows easy integration of applications software and permits data to be moved between various window processes (such as the Emulator or Editor described below).

- **VT100® Terminal Emulator (VTTY)** — permits up-loading and down-loading of data between your PC and a larger mini or mainframe computer. This function, of course, means that larger-scale development tasks can be run on more powerful machines, while permitting you to process other tasks locally.

- **TEXT EDITOR** — allows powerful, full-screen editing at the workstation level or you can use your familiar mainframe editor through the terminal emulator. Several popular editors are available.

The LANTECH SYSTEMS uNETix Software Development ToolKit is the solution to your software development needs, at an affordable price of \$399. Quantity discounts are available.

Optional Lattice® C Compilers are available from LANTECH SYSTEMS to develop programs locally to run under uNETix, or PC DOS™. Programs developed for uNETix can, of course, be easily ported to other UNIX environments. Lattice C Cross Compilers are also available for a variety of minicomputers.

To order your ToolKit, or to get more information contact:

LANTECH SYSTEMS INCORPORATED

9635 WENDELL ROAD, DALLAS, TEXAS 75243
(214) 340-4932 EX. 200

uNETix is a trademark of Lantech Systems, Inc. IBM is a registered trademark of International Business Machines. UNIX is a trademark of Bell Laboratories, AT&T Technologies. VT100 is a trademark of Digital Equipment Corp. Lattice is a registered trademark of Lattice, Inc. PC DOS is a trademark of International Business Machines. XT is a trademark of International Business Machines. Minimum system configuration is IBM PC with 512K RAM, monochrome or color display RS232 port required for VT100 connection.

Circle No. 40 on Inquiry Card

Even experienced C programmers nod at times about those two, and spend sleepless hours trying to track down the error.

My overall feeling is that the book leans a bit heavily in the direction of number crunching. In saying this, I confess that my bias is toward character crunching, the activity I observe most frequently in modern computing environments. Kelley and Pohl do venture into data structures topics, such as tree traversal, though, and I do view this as a distinct plus.

This book is a good comprehensive introduction to C if page 3, with the system dependent example program, is skipped. The table of contents follows.

Table of Contents for A BOOK ON C

0. Starting from Zero (4pp)
1. An Overview of C (12pp)
2. Syntax and the Lexical Level
3. Declarations, Expressions, Assignment, Data

Types (33pp)

4. Flow of Control (37pp)
5. Functions (26pp)
6. Branching Statements, Bitwise Expressions, and enum (26pp)
7. Pointers, Arrays, and Strings (37pp)
8. Recursion, Functions as Arguments, and the Preprocessor (28pp)
9. Structures, Unions, and typedef (27pp)
10. Structures and List Processing (29pp)
11. Input/Output and the UNIX Environment (40pp)

Appendix: ASCII Character Codes (1p)

Index to Selected Programs and Functions (3pp)

Index (8pp)

Jim Joyce is President of International Technical Seminars, Inc., a firm committed to UNIX training, and founder of the Independent UNIX Bookstore. For answers to your questions about books, call 415/621-1593.

CCA EMACS. THE MOST POWERFUL SCREEN EDITOR FOR UNIX AND VAX/VMS.

No other text editor gives you so much power, speed, and functionality as CCA EMACS™. Or makes editing so easy. Close to 400 built-in commands let you do any task with only a few keystrokes. Even things that are impossible on other editors. And with our full extension language, Elisp™, you can customize CCA EMACS to meet your program requirements.

Multiple windows is another CCA EMACS plus. So you can manage concurrent processes and move information from one window to another. And



CCA EMACS is supported by a full online documentation package that includes a novice tutorial. So any user can quickly utilize all the power of CCA EMACS.

CCA EMACS runs on Berkeley Unix™ (4.1BSD and 4.2BSD), Bell Unix (System III and System V), and VAX/VMS™ and requires 500 K of address space.

Prices for a full source license range from \$350 to \$2400.

For more information, or to find out how to get a trial copy, call Gwendolyn Whittaker at (617) 492-8860.

CCA Uniworks, Inc.

⚡ A Crowntek Company

Four Cambridge Center, Cambridge, MA 02142

Unix is a trademark of Bell Laboratories
VAX and VMS are trademarks of Digital Equipment Corporation
CCA EMACS and Elisp are trademarks of CCA Uniworks, Inc.

Circle No. 41 on Inquiry Card

See Software.

Dick is a programmer. Dick is bored. Harried. Dick struggles with trace chores. Debugging routines. Nonexistent documentation. Hidden bugs. So Dick is four months behind schedule. And customers are upset when bugs slip through. They yell and make Dick upset. They make Dick's boss upset. Nobody is very happy.



See Software Run.

Jane is a happy programmer. She uses ANIMATOR™. It's a VISUAL PROGRAMMING™ aid for MICRO FOCUS™ LEVEL II COBOL™. It runs on a micro. It makes child's play of test and debugging tasks.

With ANIMATOR Jane sees a picture of the program explaining itself. In real time. In COBOL source code. ANIMATOR tracks the program's exact execution path. Including subroutine branches. Jane can have the program run fast. Or slow. Or stop. With one key. This makes it easy to spot problems. Insert fixes. Set breakpoints. Instantly.

Jane's programs are best sellers. They're delivered on time. With no hidden bugs. Jane's boss likes this about Jane. Because he doesn't like customers to yell at him.



Run, Software, Run.

This software vendor just went public. Because he doubled productivity. Eliminated bugs. Cut costs. Produced terrific applications. Beat the competition to market. And customers don't yell at him anymore. All thanks to ANIMATOR.



See ANIMATOR now.

Let ANIMATOR help you do better work. And speed your applications to market. Write for more information. Or call (415) 856-4161. Right now.

MICRO FOCUS

2465 E. Bayshore Rd., Suite 400, Palo Alto, CA 94303

© 1984 Micro Focus Inc. All Rights Reserved.
LEVEL II COBOL, ANIMATOR, VISUAL PROGRAMMING, MICRO FOCUS and the MICRO FOCUS Logo are trademarks of Micro Focus Ltd.

2465 East Bayshore Rd., Suite 400, Palo Alto, CA 94303

I'd like more information

Name _____ Title _____

Company _____ Phone _____

Address _____

City _____ State _____ Zip _____ UR-9/84

THE UNIX GLOSSARY

Words to the wise

by Steve Rosenthal

— in most implementations of UNIX, this is the shell prompt to the superuser. It differs from other prompts so as to remind the superuser that he or she is in a special mode rather than operating as an ordinary user.

ac — the standard system command to report accounting information collected during normal system operation. The summary information can be used to charge users for CPU time and memory use.

accton — the command telling the system to keep accounting information that details when users login and out and what processes are created and killed. In most implementations, invoking **accton** (usually as **/etc/accton**) only sets the system to collecting the data. To retrieve it, use such standard programs as **prtacct** or **acctcms**, or you can write your own software to assign charges and handle billing. When **accton** is used, it is generally invoked as part of **/etc/rc** during initialization.

adduser — a command available to system administrators on some UNIX systems to add a new user to the system. The same task may be done in a less automated way by editing the **/etc/passwd** file.

bin — the login name for the user who owns the **/bin** and **/usr/bin** directories and files. Normally, this is the system operator, who is



also the superuser with login name **root**. It is considered good administrative practice for **bin** to own **/bin** and **/r/bin**.

boot device — the name of the storage unit (usually including any subdevice and block number) from which the boot program reads the UNIX kernel or any standalone test program.

chgrp — the command employed by the superuser to change the group ownership of a file or directory. In installations undergoing reorganization, this command can get frequent use.

chown — the command used by the superuser to change the ownership of a file or directory.

clri — a maintenance command employed by the superuser to remove inodes directly. In most cases, the use of this command has been replaced by **fsck**. The

task performed by **clri**, or the equivalent part of **fsck**, is needed only when the file system has become corrupted by a system crash or hardware failure.

cpio — a utility program used to copy files to and from backup devices. It is an updated version of the **tar** tape archive program.

crash — a program employed by the superuser (or in single-user mode) to look at a memory dump (core image) of the system when the system goes down. It is a utility primarily of interest to dedicated experts.

cron — a command that creates a daemon (an automatically executing program) that invokes commands at specified dates and times. The commands to be executed are listed in **/usr/lib/crontab**. The **cron** command is usually invoked as part of the **etc/rc** script executed as part of the initialization process.

date — the command the superuser uses to set the system date and time. The usual form is:

```
date yymmddhhmm.ss
```

Many systems now have real-time clocks that automatically keep and set the date and time. Ordinary users can use **date** to display the current date and time by simply entering the command without arguments.

dcheck — utility program

68000

16000

8086/88

Z8000

SOFTWARE DEVELOPMENT TOOLS "One-Stop Shopping"

OASYS provides a "One-Stop Shopping" service for software developers and managers in need of proven, cost effective, cross- and native- development tools.

OASYS can save you time, energy and money! We understand what it means to be a developer. Over the past 3 years, we've built over 1MB of working code.

We not only develop our own tools, but also specialize in evaluating, selecting and distributing the best complementary tools from other suppliers.

Our tools are currently in use in over 1,000 installations worldwide on micro-, mini-, and mainframe computers for a variety of 8-, 16- and 32- bit UNIX (and non-UNIX) systems.

Most likely, we have what you're looking for (even if it doesn't appear in the tables shown). But, if we don't, we'll be glad to tell you who does.

So, call or write today for more information and start shopping the smart way, the fast way, the economical way.

"The One-Stop Shopping Way."



60 ABERDEEN AVENUE
CAMBRIDGE, MA 02138
(617) 491-4180

CROSS TOOLS

PRODUCTS (1)	HOST (2)	TARGET (3)
C COMPILERS	VAX, PRIME	68000 16000 8086/88
PASCAL COMPILERS	VAX PDP-11, LSI-11 PRIME	68000 16000 8086/88
FORTRAN COMPILERS	VAX PDP-11, LSI-11	68000 16000 8086/88
ASSEMBLERS (4)	VAX, PDP-11, LSI-11, PRIME, IBM/PC, IBM 370	68000, 16000, 8086/88, Z8000, 680X, 808X, Z80
SIMULATORS	VAX, PDP-11 LSI-11, PRIME, IBM/PC, IBM 370	68000, 8086/88 808X, Z80

(1) WE DISTRIBUTE PRODUCTS FOR: GREEN HILLS SOFTWARE, VIRTUAL SYSTEMS, COMPLETE SOFTWARE, PACER SOFTWARE; SOFTWARE MANUFACTURERS

(2) HOST OPERATING SYSTEMS INCLUDE: VMS, RSX, RT-11, PRIMOS, UNIX V7, III, V, BSD 4.1, 4.2, UNOS, IDRIS, XENIX, MS/DOS, VM/CMS, CPM 68K

(3) OTHER TARGETS ARE: M6801-6803, 6805, 6809, 8080, 85, 28, 35, 48, 51; Z-80

(4) ALL ASSEMBLERS INCLUDE LINKER, LIBRARIAN AND CROSS-REFERENCE FACILITY

(5) AVAILABLE ON: CALLAN, OMNIBYTE, CHARLES RIVER DATA, PLEXUS, SAGE, FORTUNE, WICAT . . . to name a few.

C/UNIX NATIVE TOOLS

- NATIVE ASSEMBLERS FOR 68000s (4, 5)
- SYMBOLIC C SOURCE CODE DEBUGGER
- C-TIME PERFORMANCE UTILITY
- UP/DOWN LINE LOAD UTILITIES
- COMMUNICATION UTILITIES
- BASIC-TO-C TRANSLATOR
- C-BASED FLOATING POINT MATH PACKAGE . . . AND MORE

TRADEMARKS: UNIX IS A TRADEMARK OF BELL LABORATORIES, XENIX AND MS/DOS ARE MICROSOFT CORP'S; IBM/PC, VM/CMS, AND IBM 370 ARE INTL BUSINESS MACHINES; VAX, PDP-11, LSI-11, VMS, RSX, AND RT-11 ARE TRADEMARKS OF DIGITAL EQUIPMENT CORP; CPM 68K IS DIGITAL RESEARCH'S; PRIMOS IS PRIME'S; UNOS IS CHARLES RIVER DATA'S; IDRIS IS WHITESMITH'S LTD.

employed by the superuser (or in single-user mode) to check the integrity of the directory structure. The same function is now done at most installations by the **fsck** program. The **dcheck** program works by verifying that the number of

directories claiming an inode as a member is equal to the number of links recorded in the inode.

/dev — the directory listing all devices known to the system. To add additional devices, the superuser invokes the **mknod**

command (normally stored as **/etc/mknod**) to add the necessary information about a new device.

dump — the name of the standard system program for archiving copies of the files on disk as a hedge against system crashes or accidental file deletions. The files can be recovered from archives with the **restor** command. The **dump** utility can save an entire system of files or a subset thereof. It can be invoked on one of ten different levels. Only files modified since a dump of higher level will be saved, thus allowing selective backup. The full dump, of level 0, is known as the epoch dump.

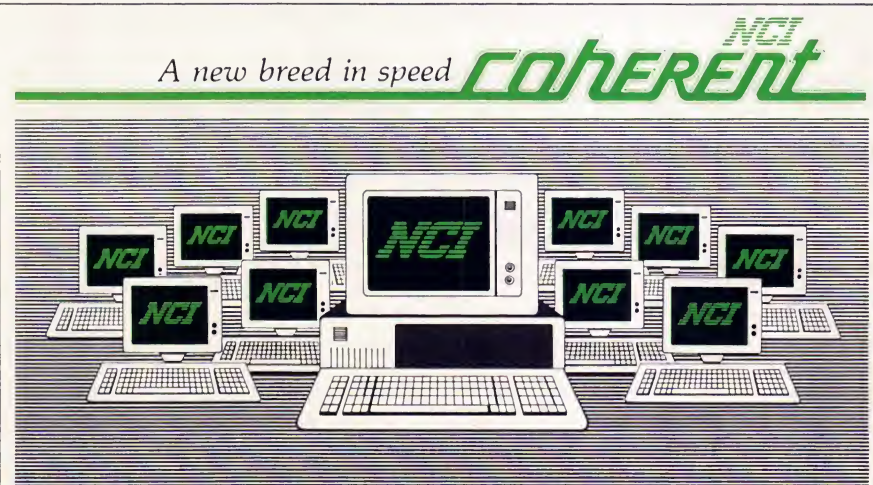
epoch — a complete dump of the UNIX system, including all the user and system files. An epoch — or level 0 — dump, is done periodically to provide a starting point for regenerating the system in case of a serious system crash.

/etc — one of the major system directories, used primarily for storing administrative programs and lists.

/etc/passwd — the system file listing user names, encrypted passwords, **userids**, **groupids**, home directories and other information about each authorized user. To add a new user, the system administrator creates a new line in the password file (either directly or by using the **adduser** command), creates a home directory for the user and gives the user ownership of the new directory.

/etc/rc — the standard file containing a shell script to be executed by **init** as part of the process of going to multiuser mode. Normally, this file contains instructions to mount directories, start daemons (system programs that run automatically) and perform such regular housekeeping as purging temporary files.

/etc/ttys — the system file specifying which terminals should be



The only UNIX compatible operating system that supports 11 simultaneous users on 1 IBM PC XT

Only NCI COHERENT offers powerful multiuser capability with UNIX compatibility.

NCI made COHERENT a multipurpose operating system. We extended it to suit a wide variety of applications. For example, it can now be used to turn a PC into a data multiplexer or an inexpensive database machine in a local area network. NCI COHERENT can also be used for high speed protocol conversion and process control. We've even written a real-time version of COHERENT, exclusive to NCI, which will run on an intelligent peripheral board.

NCI has enhanced the COHERENT kernel making it much faster. Then we added commands that make programming easier, including

a source code control system, plus many UNIX System V features.

We also gave it exclusive support for a wide variety of hardware and PC compatibles. NCI

COHERENT comes with full technical support and complete documentation, organized in the UNIX manner.

A variety of licensing options are available including runtimes, production runtimes, and driver kits.

Remember, when you license COHERENT your license includes multiuser and our low license fees

make your application far more economically feasible.

The NCI technical design team that engineered the improvements to COHERENT is also available on a contract basis.



Performance engineered into every product.

NETWORK CONSULTING INC.
Suite 110, 3700 Gilmore Way,
Burnaby, B.C. Canada V5G 4M1
Phone: (604) 430-3466

*UNIX is a trademark of Bell Laboratories. COHERENT is a trademark of Mark Williams Co. IBM PC and IBM PC XT are trademarks of International Business Machines Corp.

State of the art software and operating systems.



Tailor made consultant services.

Lachman Associates, Inc. is a custom systems software consulting firm... fifty professionals, representing two hundred man-years of UNIX experience. LAI is a system in itself; each field consultant is linked to account and project managers within the organization.

The complete UNIX resource.

We provide complete UNIX development services for hire. Our expertise also spans all of the common PDP-11, VAX and IBM/370 operating systems.

LAI has been instrumental in a number of major UNIX ports, as well as numerous projects involving:

- operating systems development
- a distributed transaction processing system
- development of a new UNIX-like operating system
- local area networks
- distributed file system research
- a UNIX front-end processor
- device drivers
- relational database systems
- compilers
- advanced debuggers
- a multi-processor implementation of UNIX
- data communications
- real-time systems
- processor architecture evaluation
- system performance measurements



- development of a heterogeneous networking system
- advanced graphics software
- product analysis

Our related services include market studies, customized technical training, technical documentation and software research and development.

Human engineering.

We tailor our skills and experience into an understanding of client requirements. With this knowledge, we tailor projects completely from providing contract programmers to supplement your existing staff to total project responsibility, including system specification, design, implementation, acceptance testing and training.

L A C H M A N

Lachman Associates, Inc.

Corporate Offices
645 Blackhawk Drive
Westmont, IL 60559
312 986-8840

Chicago Denver New Jersey

the UNIXTM people

UNIX is a trademark of Bell Laboratories. PDP-11 and VAX are trademarks of Digital Equipment Corporation

In the market for UNIX™
hardware? software?
perhaps a whole system?
Maybe you're not sure?
In any case
remember

RULE #1: USERS KNOW BEST!

(with a little help)

Whatever you're about to buy, it's always smart to talk to someone who already uses it.

At B.A.S.I.S. we routinely use a score of UNIX™ software and hardware configurations. Our staff can tell you about the strengths, weaknesses, and quirks of our hardware and software products...from a user's perspective.

Remember, whether you need a big system, a little system, or just a piece of a system, ask someone who uses
big systems,
little systems,
and all the pieces:

B.A.S.I.S.

B.A.S.I.S.
"The UNIX™ Users"
1700 Shattuck Avenue
Suite 1
Berkeley, CA 94709
(415) 841-1800

UNIX™ is a trademark
of Bell Laboratories.
Circle No. 46 on Inquiry Card

UNIX PROFESSIONALS

Our clients, nationally based, are seeking software specialists, systems engineers, and programmer analysts for interesting projects both corporate and consulting. Excellent benefits, salaries range up to \$70,000.

We are an executive search firm, specialists in the growing computer industry and marketing development to service UNIX professionals.

**PLEASE CALL OR SUBMIT
A RESUME IN CONFIDENCE TO:**

N.M. PRO DATA LTD.
172 MADISON AVENUE,
SUITE 304
NEW YORK, N.Y. 10016
212-679-7966

ATTN: MARLENE SAFERSTEIN
Circle No. 47 on Inquiry Card



Q'Nial

The new, sophisticated,
interactive programming system
for
UNIX Workstations
Super Minicomputers (VAX)
and the
IBM Personal Computer

Suitable for:
Artificial Intelligence
Rapid Prototyping
Data Systems Design

NIAL Systems Limited
20 Hatter St.,
Kingston, Ontario
Canada K7M 2L5
1-613-549-1432
1-800-267-0660 (U.S.)

Q'Nial is a registered trademark
of Queen's University at Kingston

checked for logins and further instructions. Along with specifying active terminals, the list provides terminal numbers and a method for establishing baud rate (terminal speed).

find — the command for locating files in directories and subdirectories matching specified criteria. It can find files by name, owner, mode, age, size and several other criteria. It is often used by the system administrator when file space grows tight or runs out to see if anyone is hogging space or if there are files that have not been used recently that can be archived offline.

fsck — a command used for checking the integrity of the root file system or any other file system specified in the file `/etc/checklist`. This command checks for duplicate or missing links, bad blocks, an incorrect free list, and other defects. In most cases, when **fsck** finds an error, it will suggest a fix and ask for confirmation.

fsdb — a file system debugger program used by people who feel competent to manually reconstruct file systems or alter disk records. Though an alternative to **fsck** for fixing the file system after a crash, **fsdb** requires a more detailed knowledge of the system.

icheck — a utility program that checks the integrity of the inodes and free block list of a file system. The same task is now done at most installations by **fsck**. These programs are mostly used when there is some cause to suspect a problem in the file system, such as after a system crash or hardware error.

logbook — a record of the actions taken by the system operator along with status information from the system. At large installations, especially those with multiple operators, keeping a log is essential to maintaining orderly backups and keeping the system running smoothly.

The Right Machine.

68000

The Right Operating System.

UNIX

The Right Language.

Pascal-2™

Perfect Timing.

**Oregon
Software**

6915 S.W. Macadam Avenue, Portland, Oregon 97201

For Technical Information and Price, Call Toll-Free:

1-800-874-8501

In Oregon, Call: (503) 226-7760

UNIX is a trademark of Bell Laboratories.

Circle No. 49 on Inquiry Card

mkfs — the command used by the system administrator to create a new file system. After a file system is created, it must be mounted before it becomes accessible to the system.

mknod — the command employed by the superuser to add devices to the system. The process, called "making a node," adds appropriate entries to the /dev system directory.

mount — to add a file system to a directory in the existing root file system. This process is usually done by the superuser, either during system startup or when disks or tapes are changed on removable-media devices. A new file system must be created with **mkfs** before it can be mounted for

the first time, and if it is to be associated with a new directory, that entry must first be created with **mkdir**.

multiuser — the normal state of a UNIX system, allowing more than one user to run jobs at various consoles. When UNIX is first booted up, however, it normally comes up in single-user mode to allow the system operator to perform housekeeping and system maintenance.

passwd — a command that the superuser can employ to change anyone's password. Some administrators supply new users with an initial password, but others leave the password initially blank and let users pick their own. However, the supervisor may fre-

quently be called upon to use this command to assign a new password to users who have forgotten their existing ones. Normal users can also use **passwd** to change their own passwords.

real — when referring to system timing (as, for example, reported by the **time** command), "real" refers to absolute elapsed time — as it is seen from the outside world. This is also called "wall-clock" time.

root — the user name for the superuser, so called because he or she "owns" the **root** directory. Some system commands can only be executed by **root**. Also, **root** serves as the name of the highest directory in the file system, written /.

MIPS SOFTWARE PROVIDES THE APL - UNIX® SOLUTION: DYALOG APL

UNIX® based - fully functional commercial APL including nested arrays, upper and lower case data support for the UNIX® environment, dynamic workspace size, external functions (callable subroutines written in other languages) full screen editor, error trapping, commercial formatter and a host of other desirable features.

DYALOG APL is available for a variety of UNIX® computing environments including VAX®, PE, Gould, 3 B Series!, NCR Tower, Zilog, Fortune, Perq, Ridge!, Pyramid!, and Sun!. For further information about DYALOG APL on your 68000, 16032 or 8086 base system, call or write today.

MIPS SOFTWARE DEVELOPMENT, INC.
31555 West 14 Mile Road
Suite 104
Farmington Hills, MI 48018
313-855-3552

UNIX® is a trademark of Bell Laboratories
VAX® is a trademark of Digital Equipment Corporation.
! Call for Availability

α ω τ ρ α λ ε ι ο

Circle No. 50 on Inquiry Card

C COMPILER

- FULL C
 - UNIX* Ver. 7 COMPATABILITY
 - NO ROYALTIES ON GENERATED CODE
 - GENERATED CODE IS REENTRANT
 - C AND ASSEMBLY SOURCE MAY BE INTERMIXED
 - UPGRADES & SUPPORT FOR 1 YEAR
- C SOURCE AVAILABLE FOR \$2500⁰⁰

HOST	6809 TARGET	PDP 11* LSI 11* TARGET	8080 (Z80) TARGET	8088 8086 TARGET
FLEX* UNIFLEX* OS-9*	\$200.00 \$350.00	500.00	500.00	500.00
RT 11* RSX 11* PDP 11*	500.00	200.00 350.00	500.00	500.00
CP/M* 8080 (Z80)	500.00	500.00	200.00 350.00	500.00
PCDOS* CP/M86* 8088 8086	500.00	500.00	500.00	200.00 350.00

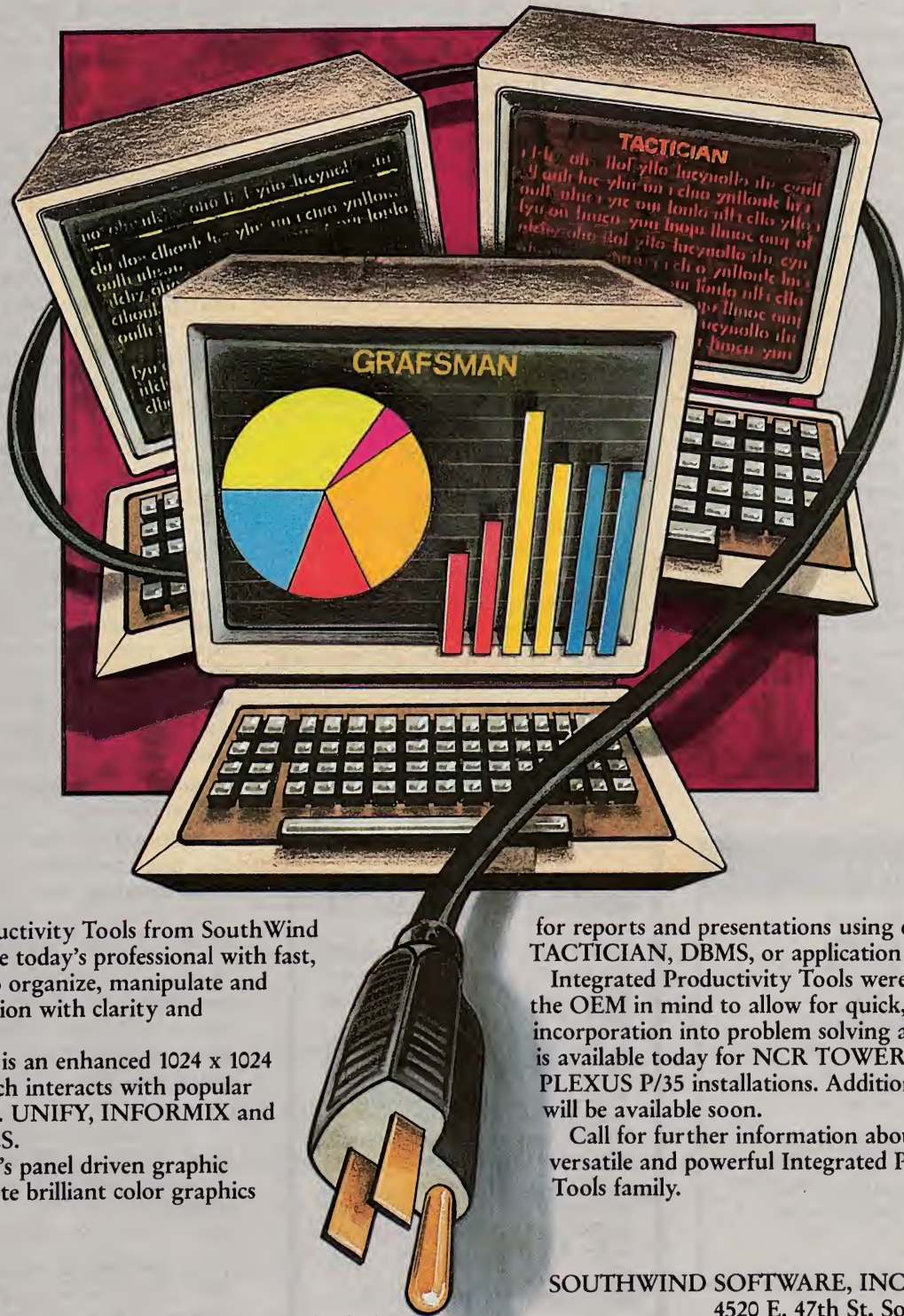
*PCDOS is a trademark of IBM Corp. MSDOS is a trademark of MICROSOFT. UNIX is a trademark of BELL LABS. RT 11/RSX 11/PDP-11 is a trademark of Digital Equipment Corporation. FLEX/UNIFLEX is a trademark of Technical Systems consultants. CP/M and CP/M86 are trademarks of Digital Research. OS-9 is a trademark of Microware & Motorola.

408-275-1659

TELECON SYSTEMS
1155 Meridian Avenue, Suite 218
San Jose, California 95125

Circle No. 51 on Inquiry Card

Fast...Versatile ...Powerful Tools



Integrated Productivity Tools from SouthWind Software provide today's professional with fast, versatile tools to organize, manipulate and display information with clarity and effectiveness.

TACTICIAN is an enhanced 1024 x 1024 spreadsheet which interacts with popular UNIX DBMS ie. UNIFY, INFORMIX and MICROINGRES.

GRAFSMAN's panel driven graphic editor helps create brilliant color graphics

for reports and presentations using data from TACTICIAN, DBMS, or application files.

Integrated Productivity Tools were designed with the OEM in mind to allow for quick, easy incorporation into problem solving applications. IPT is available today for NCR TOWER 1632 and PLEXUS P/35 installations. Additional IPT ports will be available soon.

Call for further information about the fast, versatile and powerful Integrated Productivity Tools family.

SOUTHWIND SOFTWARE, INC.
4520 E. 47th St. So.
Wichita, Kansas 67210
316-788-5537
1-800-346-3025 EXT 234



Integrated Productivity Tools, IPT, TACTICIAN, and GRAFSMAN are trademarks of SouthWind Software, Inc. UNIX is a trademark of AT&T Bell Laboratories. UNIFY is a trademark of the Unify Corporation. INFORMIX is a trademark of Relational Database Systems, Inc. MICROINGRES is a trademark of Relational Technologies, Inc. NCR TOWER 1632 is a trademark of the NCR Corporation. PLEXUS P/35 is a trademark of Plexus Computers, Inc.

Circle No. 52 on Inquiry Card

shutdown — a command on many newer UNIX systems that automatically runs through the steps needed to shutdown the system. It notifies all current users, synchronizes files (writes buffers and superblocks to disk), and brings the system back to single-user mode.

single-user — the mode in which UNIX first comes up when booted (started up). The single user has the power of *root*, the superuser. After the superuser initializes the system, performs any maintenance and kills the single-user process, the system switches to multiuser mode.

/stand — a special directory in many UNIX systems containing standalone versions of various programs to check hardware and file system integrity. These programs can be called directly from the boot program, before the system is brought up.

su — the command to switch user. It allows you to temporarily login as another user (providing you know the other user's password). When you logout from

that other user's account (normally by responding to that user's shell prompt with a CTRL-D), you return to your previous account. One of the main uses of this command is to allow the administrator to login first as an ordinary user, changing to superuser mode only when necessary. This practice is a safety measure, since the superuser can override protections and might do so inadvertently. Most systems, therefore, encourage superusers to do normal work in ordinary user mode.

superuser — the general term for the UNIX system operator. In UNIX, the operator, who has the login name *root* and userid 0, can override most protections, kill processes, mount and unmount file systems, change passwords, and do many other things that ordinary users cannot do. Thus comes the term "superuser." At most installations, the superuser has another username for ordinary access to the system and only uses the superuser mode when bringing up the system or when the special superuser powers are required.

sync — the command to flush buffers and the in-core superblock to disk. The superuser should issue this command before unmounting (taking off line) a storage device or before shutting down the system.

umount — the command employed by the superuser to logically remove a file system from the directory that contains it — a step that is necessary prior to physically removing or disconnecting a storage device. The device to be unmounted must not contain the current working directory of any active user, and the **sync** command should be issued immediately beforehand to flush any buffers containing information for the device.

update — a daemon (independently executing program) invoked during initialization to regularly dump file buffers to disk by issuing the **sync** system call. With some hardware, there is a slight danger that **update** can actually corrupt stored files if updating occurs during a system crash, but most users feel that the advantages of automatic updates make them worth the risk.

user — as applied to timing measurements, "user" refers to the amount of time spent running the **user** program. This may be considerably less than elapsed time (real time), especially on a heavily-loaded system.

wall — the command that writes messages to all current system users. Its most common use is to warn users that the system will be shutdown at a certain time.

If you have comments, please send them to Rosenthal's UNIX Glossary, Box 9291, Berkeley, CA 94709.

Steve Rosenthal is a lexicographer and writer living in Berkeley. His columns regularly appear in six microcomputer magazines. ■

UNIX*

Training in
DALLAS,
Austin and Houston
by
TELOS

1 Day Overview..... \$145
3 Day Fundamentals
with hands-on..... \$545
Call or Write for Schedule.

TELOS CONSULTING SERVICES
833 E. Arapaho, Suite 106
Richardson, TX 75080

*UNIX is a trademark of Bell Laboratories.

Circle No. 53 on Inquiry Card

THE SOLUTION

24-hour, UNIX System III timesharing via Telenet. As low as \$5.95 per hour connect + \$.03/cpu sec nonprime. No additional charge for 1200 baud.

C, FORTRAN77, PASCAL, SNOBOL, RATFOR, COBOL, BS, ASSEMBLER, and LISP

USENET Bulletin Board System typically brings you over 160 new articles per day in 190+ newsgroups from a network of over 800 UNIX sites worldwide.

\$24.95 brings you 1 hr. FREE system time + BYTE BOOK Introducing The UNIX System by Morgan & McGilton (556 pp.) + Solution News subscription.

Korsmeyer 5701 Prescott Avenue
ELECTRONIC DESIGN, INC. Lincoln, Nebraska 68506-5155
402-483-2238

Circle No. 54 on Inquiry Card

The Software Professionals' JOB FAIR

SOFTFAIR[®]

BOSTON AREA

84

September 17 & 18, 1984
4pm-9pm both days
Hillcrest Function Center
Waltham, MA

October 1 & 2, 1984
3:30pm-8:30pm both days
Park Plaza Hotel
Boston, MA

These companies made SoftFair '84 a success. They saw SoftFair as a tremendous vehicle to meet experienced software professionals.

• Accupoint • ADE Corporation • Apollo Computer • Atex • Bank of Boston
• Barry Controls • BBN Communications • Berklee College of Music • Beth
Israel Hospital • BKW, Inc. • Blue Cross/Blue Shield • Boston Stock Exchange
• Bradford Trust Company • Bunker Ramo Information Systems • Carleton
Corporation • Charles Stark Draper Laboratory • Codex • Cole Surveys • Com-
munications Analysis • Compugraphic Corporation • Computer Corporation of
America • Computervision • Cullinet Software • Data General • Digital
Equipment Corporation • Dynamics Research • The Faxon Company • The
Federal Reserve Bank of Boston • The Foxboro Company • GCA • G.E.
Information Services • GENESYS Software Systems • Genrad, Inc. • Gould,
Inc. • Graphic Communications, Inc. • Higher Order Software • Hills Depart-
ment Stores • Honeywell • Infocom, Inc. • Interlan • Interleaf • IOCS • John
Hancock Life Insurance • Leading Edge • LFE Corporation • Liberty Mutual
• LTX Corporation • Masscomp • McCormack & Dodge • The Mitre Corpora-
tion • NEC • New England Tech Writing Associates • Ovation Technologies
• Prime Computer • Raytheon Missile Systems Div. • Raytheon Service Com-
pany • RTC Systems • Sanders Associates • Sequoia Systems • Shawmut
Bank of Boston • Softrend • Software International Corporation • Software
Research Corp. • System Development Corporation • Telelogic • Textet • Turning
Point Systems • United Brands Company • Vitro Corporation • Wang Labs

People like you also made SoftFair a success because they saw SoftFair '84 as a way to get a broad view of the software job market without writing lengthy cover letters, mailing numerous resumes or making inconvenient phone calls.

This Fall SoftFair '84 returns to Boston. Many of Boston's top-flight companies will be at SoftFair '84 offering exciting career opportunities to **experienced** software professionals in a variety of **Software Engineering, Scientific and Business Applications** areas related to: DEVELOPMENT, SUPPORT, IMPLEMENTATION, TRAINING, QUALITY ASSURANCE, TECHNICAL WRITING, SALES, MARKETING, and MANAGEMENT. Come in, meet them, and get a real good view of the New England marketplace.

If you can't attend SoftFair '84, send a resume with cover letter, in complete confidence, stating which companies you'd like to have receive your resume to: Dave Callahan, Software Career Link, Dept. UR9, 67 South Bedford Street, Burlington, MA 01803, or for more information call (617) 229-5813.

SOFTFAIR '84 is sponsored by **SOFTWARE CAREER LINK** serving equal opportunity employers.



Continued from Page 60

and a determination on their part to assess the risks and cause the right actions to take place. Sec-

only, they will need to institute and disseminate proper administrative control in accordance with

their findings. Thirdly, employee education as to what is expected of them is necessary.

REVIEW: *What are the measures you use?*

MORRIS: We do some of that. I think it's just that we could do better. We do have quality assessment. My management is aware of the problem. I talk to them about it, and I see what they do. Some administrative policies and techniques have been set up and have been enforced. There has been at least some genuine attempt to make the employees aware of the risks involved and what's expected of them. We tell people good ways and bad ways to choose login passwords. We tell them the kind of information we expect them not to put on dialup computers.

REVIEW: *So how much security is enough?*

MORRIS: You find out where your exposures are, what the risks are, make some kind of actuarial assumptions about the costs to assign to the risks, and treat them as if they were risks you were trying to insure against — ones that you might have to pay a premium for. If it's a bet-your-company situation, you had best take strong steps and spend a lot of money to really protect that information.

Here's a typical cost, a cost that I've looked at for an international company with about 20 branches that has a high flow of information all day long, seven days a week. They asked me to estimate how much it would cost to really protect that stuff against pretty determined efforts. It turns out that the answer for that whole network is about \$1,000,000 of investment.

That's because this company is in the communications business, fundamentally, rather than in just a data processing situation. They were mainly interested in the communications aspect of it.

NEW

ONE SIZE FITS ALL

Heurikon presents Minibox — a multiuser UNIX workstation based on its powerful HK68™ single board microcomputer and Uniplus+™ UNIX System III or System V operating system with Berkeley enhancements.

Designed with the OEM in mind, *one size fits all*. Both compact and flexible, the Minibox includes within its 10.5" w x 13.9" h x 20.5" f frame a 200 or 400 watt power supply, six slot Multibus™ card cage, (4-5 available for user use!), single double density floppy disk drive, streamer tape drive, and 31 or 65 Mbyte Winchester drive (expandable to 280 Mbytes). All this within the same cabinet! System status LEDs on the front panel inform the user of CPU and disk drive activity.

With Uniplus+™, Minibox becomes a flexible and affordable tool for program development, text preparation, and general office tasks. Included is a full "C" compiler, associated assembler and linker/loader. Optional languages are:

Macro assembler, ISO Pascal compiler, FORTRAN-77 compiler, RM-COBOL™, SVS BASIC (DEC BASIC compatible interpreter), SMC BASIC (Basic-Four BB3 compatible interpreter), and Ada™. Other utilities include UltraCalc™ multiuser spread sheet, Unify™ DBM, Ethernet™, and floating point processor. Alternate operating systems available are PolyForth™, Regulus™, CP/M 68K™, and others.

*UNIX is a trademark of Bell Laboratories. Unify is a trademark of Unify Corp. UltraCalc is a trademark of Olympus Software. Ethernet is a trademark of Xerox Corp. Uniplus+ is a trademark of UniSoft Corp. PolyForth is a trademark of Forth, Inc. Regulus is a trademark of Alcyon Corp. CP/M-68K is a trademark of Digital Research. Ada is a registered trademark of the U.S. government, Ada Joint Program Office. RM-COBOL is a trademark of Ryan-McFarland Corp. HK68 is a trademark of Heurikon Corp. Multibus is a trademark of Intel Corp.

MINI BOX
Mini Micro Computer Booth #2215-17

HEURIKON CORP
© 1983

3201 Latham Drive
Madison, WI 53713
Telex 469532

800/356-9602
In Wisconsin
608/271-8700

Circle No. 56 on Inquiry Card



FUSION™ is the only connection.

In an industry fragmented by diversity, FUSION makes connections.

FUSION is the LAN software that links different operating systems, diverse LAN hardware, and diverse protocols, to give you high speed communication across completely unrelated systems.

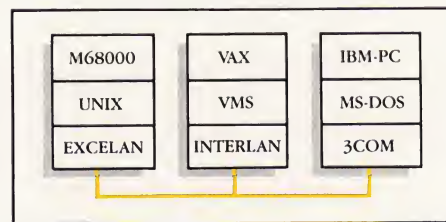
FUSION frees you to buy the best computer for the job—without worrying about compatibility. Your VAX mainframe can now communicate with your IBM-PC. You can even add M68000 work stations or a communications server. FUSION links them all—and more. Regardless of operating system, protocol, or network hardware.

FUSION™

- With FUSION, any user on any system can perform file transfer, remote login, and remote execution.
- FUSION offers a choice of two standard protocols—XNS or TCP/IP. Some customers have even developed private protocols.
- And you can create your own application programs, using FUSION utilities.
- FUSION accommodates all major LAN boards, too. So network interface is no obstacle.

When you add FUSION, you never lose your hardware investment. In fact, you get more out of it than ever.

Ask for FUSION. It's the connection you've been looking for.



A sample network using a few of the many configurations made possible with FUSION.

For further information please contact:
Network Research Corporation
1101 Colorado Ave.
Santa Monica, CA 90401
(213) 394-7200
(800) 541-9508

FUSION is a licensed trademark of Network Research Corporation.



See us at UNIX SYSTEM EXPO/84, Los Angeles, September 12-14, Booth #103



The computers were really only storage organs to serve a communications facility. The indirect costs were trivial. It was primarily a question of upfront capital

cost. Compared with the size of the company, this cost was absolutely trivial, just lost down in the noise level. The direct cost of even very good protection is not

particularly high.

REVIEW: *What do you do when breaches occur?*

MORRIS: We have an organization within the company called the Assets Protection Organization where, theft of information is treated in exactly the same way as theft of physical assets. When a breach occurs, it's routinely reported to the Assets Protection Organization. Things have advanced to the point where computer break-ins are treated just like any other theft or interference — illegal acts that are treated as such.

REVIEW: *Do you think many of these breaches are malicious or not?*

MORRIS: I have never myself encountered a genuinely malicious act, but I've read about them in the newspapers.

I think the motivation for most of the kind of stuff you read about in the newspapers is one of two things: it's either high school kids playing games or it's crooks stealing.

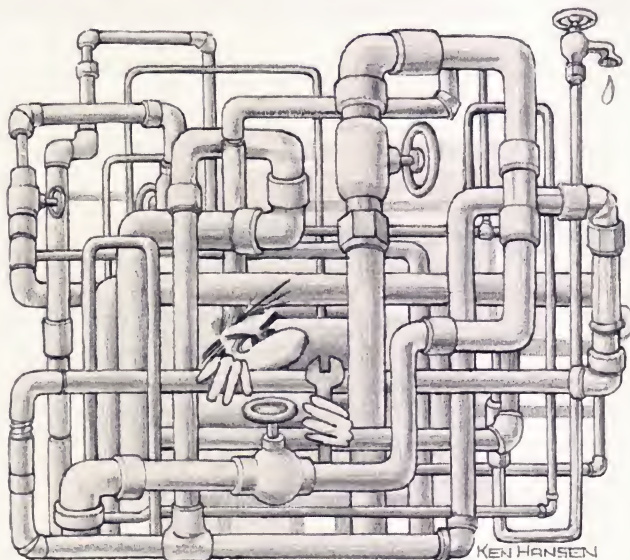
REVIEW: *And even there, they're not particularly inclined to be destructive to other things; they're just trying to steal what they're trying to steal.*

MORRIS: Certainly. I think they just want to steal the stuff and get out.

REVIEW: *Would you say a little bit about the impact of UUCP on system security?*

MORRIS: Sure. I think UUCP makes the problem wider. First off, anything that makes access to a computer easier makes the security problem more difficult. Secondly, there is a tendency for the security of the networks — that is, of every computer on the system — to be that of the weakest.

This brings up another important administrative problem. If



Without SoftShell™, learning to live with UNIX™ could be a real trap.

SoftShell is a convenient interface which guides you gracefully through UNIX. If you're new to UNIX, or have the task of training new users, SoftShell simplifies its complexity. If you're already a UNIX fan, you'll love SoftShell because it augments your system. You'll find yourself further exploring the great depth and versatility of UNIX. Available for UNIX System-V, Berkeley 4.2 and IBM's PC/IX, SoftShell is an invaluable tool for users at all levels of expertise. For information, contact Logical Software, Inc., 17 Mount Auburn Street, Cambridge, MA 02138, (617) 864-0137.

UNIX is a trademark of AT&T Bell Laboratories.



Logical Software Inc.

See SoftShell at UNIX EXPO in New York City, Oct.16-18, Booth #617

Circle No. 44 on Inquiry Card

you're going to admit other computers onto the common network, then you have a stake in their security. If all the machines are within the same company, for example, the administration is going to have to insist that all those computers come under some known standards.

REVIEW: *In order to rationally approach the problem, UUCP in particular was distributed for a long time with permissions set so that people from remote sites could copy files from the host system. I gather that arrangements are being made for future releases of UUCP to be sent out in a form that is more restrictive.*

MORRIS: I'm not sure what form it's being sent out in. The only thing I know for sure is that the 20 or 30 inhouse computers I have some contact with cannot be persuaded from a remote place to send you any file at all if there was not a previous arrangement. They send files only from one particular directory, so that in order to let someone have a file, you would first have to put a copy of it in that directory.

REVIEW: *One concern that's been expressed to me relates to Ethernet and other local area networks. In general, if passwords are sent over such a network, one machine with a promiscuous listening mechanism would be able to gather the passwords of anybody who signed on while it was listening.*

MORRIS: Oh yes, sure. But you assume in an Ethernet that you've got some control over who's on the net. I mean, if you don't and there's promiscuous access to a local area network, all hell breaks loose. You've just got your stuff lying on the sidewalk. I don't know of any situation like that.

There is one other subject that's related both to UUCP and the local area network. That is, in

both of these situations, what really happens is that the computer gets a message that it trusts is true. This confidence is sometimes unwarranted. In all of these situations, fraudulent forged messages are possible.

REVIEW: *This is another mechanism that should be classified with Trojan Horse programs.*

MORRIS: Not a bad idea. Because, for example, with UUCP, you cannot find out who calls you up. But you can find out who they

Things have advanced to the point where computer break-ins are treated just like any other theft or interference.

said they were. So I can write up my own version of a UUCP program, run it, and thus tell the machine I call up that I'm somebody else. They'd have a heck of a job trying to get beyond that information. So you get a message and you place a measure of trust in it because you estimate what the risks are in leaving the message.

REVIEW: *I have one major question left, and it's the most open-ended of the set. What do you see as the future of security management?*

MORRIS: As time goes on, I think we're going to see increased

awareness and perception of the risks and exposures. I think that's going to happen faster than any really important technical developments. I doubt, for example, that encryption will become a very big part of all this, because encryption has been around for a long time. So instead of relying on encryption, I think that when people begin to estimate costs, there will be a strong impetus to write better software. I think a program of the general nature of UUCP will be written with specific requirements laid out in advance.

There's a strong, growing interest in information protection. Everywhere around I see the subject getting more and more attention at all levels — higher levels than in the past. So I am fairly optimistic.

REVIEW: *Are we seeing it show up, for example, in MBA programs in the various universities?*

MORRIS: Oh, yes. I give talks all the time. I've given six or eight talks this year alone — government and nongovernment, industrial, university and conference audiences.

I very rarely accept an invitation from an outside company. The last time I did, it was a very large company, and sitting in on my talks was the president of the company and every vice president.

REVIEW: *The point is: they had already come to the same conclusion you had. That is, they already saw that the problem was largely an administrative one.*

MORRIS: Completely. I was very pleased with that. Those people are on the right track, and they will have no problems at all. I could have walked out at the beginning of my talk and simply said, "Hey, just the fact that you guys are here means that you



know what the problem is and how to solve it.”

Security is not strictly a matter of administrative policy, though. I would like to talk about what I think is the most important of all the technical solutions put into place — the notion of the *auto-trace*. I consider that supremely important. Whatever software you have on a computer that attempts to foil unauthorized disclosure might work, or it might not work. If it doesn't, you've perhaps lost some information, but what is

**Computer and
information
security is primarily
an economic
problem.**

characteristically much more important is that you may have lost something and not even know you've lost it. But if you put a lot of investment in setting up audit trails, you can at least find out when something has happened, and find out from the records how it happened. In that way, you can at least close the door after the horse is gone, which is a whole lot better than not closing the door at all.

REVIEW: *Or not even noticing that the horse is missing.* ■

Coming up in October

- Is it still UNIX?
- Front-end schemes.
- Measures for compatability.
- Reasons behind mainframe interest in UNIX.
- The CAE connection.
- Historical perspectives.

THE INDEPENDENT UNIX* BOOKSTORE

OVER 60 UNIX AND C ITEMS IN STOCK INCLUDING

■ UNIX BOOKS

- The UNIX System by Stephen R. Bourne
(Addison-Wesley)
- Operating System Design: The XINU Approach
by Douglas Comer (Prentice-Hall)
- The UNIX Programming Environment by Brian
Kernighan and Rob Pike (Prentice-Hall)
- A Practical Guide to the UNIX System by
Mark G. Sobell (Benjamin/Cummings)

■ C BOOKS

- The C Puzzle Book by Alan R. Feuer
(Prentice-Hall)
- The C Programming Language by Brian W.
Kernighan and Dennis M. Ritchie (Prentice-Hall)
- Learning to Program in C by Thomas Plum
(Plum-Hall)
- C Programming Guide by Jack Purdum
(Que Corp.)

■ C and vi REFERENCE CARDS

■ PERIODICALS

- UNIQUE
- UNIX Review
- World UNIX & C

■ vi POSTER, UNIX SHELL POSTER

■ T-SHIRTS

- "UNIX is a Trademark of Bell Laboratories"
- "-rwxrwxrwx"
- "grep for it"
- "awk: bailing out near line 1"

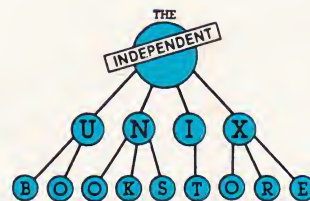
■ NEW ITEM

- UNIX Coffee Mug
- "UNIX" in blue on outside
- "is a Trademark of Bell Laboratories"
- inside lip imprint
- Porcelain mug, kiln-fired permanent color

Call or write for a complete catalog

Mail and phone orders only.
We ship anywhere in the U.S.
Contact us for shipping elsewhere.

* UNIX is a Trademark of Bell Laboratories



A Division of

International Technical Seminars
520 Waller Street
San Francisco, CA 94117
(415) 621-1593

CALENDAR

SEPTEMBER

September 11-14 UNIX Systems Expo/84, Los Angeles, CA. Contact: Computer Faire, Inc., 611 Veterans Boulevard, Redwood City, CA 94063. 415/364-4294, or CFI in Newton, MA, 617/965-8350.

September 19-21 UNIX Systems Exhibition 84, Cambridge, England. Contact (Organized for the EUUG and /usr/group by): Network Events Limited, Printers Mews, Market Hill, Buckingham, MK181JX, England. (0280) 815226.

OCTOBER

October 2-4 East Coast Computer Faire, Boston, MA. Contact: Computer Faire, Inc. (see September 11-14).

October 16-18 UNIXEXPO, The UNIX Operating System Exposition, New York, NY. Contact: National Expositions Co., Inc., 14 West 40th Street, New York, NY 10018. 212/391-9111.

October 25-28 The Second PC Faire, San Francisco, CA. Contact: Computer Faire, Inc. (see September 11-14).

NOVEMBER

November 14-18 Comdex, Las Vegas, NV. Contact: The Interface Group, Inc., 300 First Avenue, Needham, MA 02194. 617/449-6600.

JANUARY 1985

January 22-25 UniForum, The International Conference of UNIX Users, Dallas, TX. Contact: UniForum/Professional Exposition Management Co., 2400 East Devon Avenue, Des Plaines, IL 60018. 800/323-5155.

TRAINING CALENDAR

SEPTEMBER

September 5 AT&T Technologies, Hopewell, NJ: "UNIX System Screen Editor vi." Contact: AT&T Technologies, Corporate Education & Training, PO Box 2000, Hopewell, NJ 08525. 800/221-1647.

September 5 AT&T Technologies, Sunnyvale, CA: "Fundamentals of the UNIX Operating System for Programmers." Contact: AT&T (see September 5).

September 5-7 CAPE Seminar, Austin, TX: "A User-Oriented Evaluation Three-Day Seminar: UNIX." Contact: Center for Advanced Professional Education, 1820 East Garry St., Suite 110, Santa Ana, CA 92705, 714/261-0240.

September 5-7 Digital Seminar Program, New York, NY: "Comprehensive Overview of the UNIX Operating System." Contact: Digital Educational Services, 12 Crosby Dr., Bedford, MA 01730. 617/276-4949.

September 6 AT&T Technologies, Lisle, IL: "Overview of the UNIX System." Contact: AT&T (see September 5).

September 10 AT&T Technologies, Sunnyvale, CA: "Shell Command Language for Programmers." Contact: AT&T (see September 5).

September 10 AT&T Technologies, Lisle, IL: "Fundamentals of the UNIX Operating System for Users." Contact: AT&T (see September 5).

September 10 AT&T Technologies, Hopewell, NJ: "UNIX System Document Preparation." Contact: AT&T (see September 5).

September 10 NCR Education Seminars, Dallas, TX & Dayton, OH: "UNIX Operating System." Contact: NCR Education Center, 101 West Schantz Ave., Dayton, OH 45479. 800/841-CASE, or in Ohio, 800/845-CASE.

September 10 NCR Education Seminars, New York, NY: "C Programming." Contact: NCR (see September 10).

September 10 Pulsetrain Seminar, New York, NY: "PROLOG Programming Seminar." Contact: Pulsetrain, Attn: John Malpas, 747 Greenwich St., New York, NY 10014. 212/255-2385.

September 10-12 Computer Technology Group, San Francisco, CA: "UNIX Fundamentals for Non-Programmers." Contact: CTG, Telemedia, Inc., 310 S. Michigan Ave., Chicago, IL 60604. 800/323-UNIX or in Illinois, 312/987-4082.

September 10-14 Bunker Ramo Information Systems, Trumbull, CT: "Advanced C." Contact: Bunker Ramo Information Systems, Training Services Group, Trumbull Industrial Park, Trumbull, CT 06609. 203/386-2223.

September 10-14 Structured Methods Seminar, New York, NY: "UNIX System Workshop." Contact: Structured Methods, Inc., 7 West 18th St., New York, NY 10011. 800/221-8274, or in NY, 212/741-7720.

September 10-14 Uniq Digital Technologies Seminar, Chicago, IL: "The UNIX Operating System." Contact: Uniq Digital Technologies, Attn: Dennis Meyer, 28 South Water St., Batavia, IL 60510. 312/879-1566.

September 11 Computer Technology Group, New York, NY & Washington, DC: "UNIX Overview." Contact: CTG (see September 10-12).

September 12-14 CAPE Seminar, Philadelphia, PA: "A User-Oriented Evaluation Three-Day Seminar: UNIX." Contact: CAPE (see September 5-7).

September 12-14 Computer Technology Group, New York, NY & Washington, DC: "UNIX Fun-

Everything You Need To Know About UNIX . . .

But Don't Know Where To Ask



The Unix Operating System Exposition & Conference

October 16, 17, 18, 1984

Sheraton Centre Hotel—
Conference

Marina Expo Complex—
Exposition

UNIX EXPO

The comprehensive, practical business/learning event designed solely and specifically to address the myriad business and technical aspects of the UNIX OPERATING SYSTEM. UNIX EXPO is *the* national trade show that will bring ISO's, sophisticated end-users, technical personnel, OEM's, software dealers, and other resellers face-to-face with the leading suppliers to the industry at the exposition, and the leading UNIX authorities at the conference program. By attending this three day forum, you can be prepared to position yourself at the vanguard of the UNIX revolution.

Learn To Earn at the Conference Program

A penetrating, multi-track slate of seminars focusing on *the* most vital technical and business areas of UNIX has been developed by noted UNIX advocate, James Joyce, President, International Technical Seminars. Attending the conference will help you achieve a full understanding of what is destined to be the major computer operating system for the coming decades.

Inspect - Compare - Question - Select

all of the UNIX products and services on display at the 350 booth exposition. The nation's leading suppliers of UNIX and UNIX-like hardware, software, peripherals and services are anxious to talk business with you.

Meet the Leaders in the Expanding UNIX Universe

For three days in October, New York City, the heart of the largest computer marketplace in the world, will become the core of the UNIX universe; creating an unparalleled opportunity for you to meet and exchange ideas, theories and information with your colleagues.

Expand Your Horizons At the Job Fair

PENCOM SYSTEMS, the national recognized leader in UNIX recruiting will host a special JOB FAIR at UNIX EXPO where exhibiting firms will disseminate information regarding employment opportunities. Your career objectives can be discussed, and meetings with company representatives scheduled.

UNIX EXPO

Return to:
National Expositions Co., Inc.
14 W. 40 St.
N.Y., N.Y. 10018

- I am interested in attending UNIX EXPO.
 I am interested in exhibiting in UNIX EXPO.
Please send me full details.

Name _____

Title _____

Company _____

Address _____

City _____

State _____ Zip _____

Or Call: 212/391-9111, for immediate information

Want All The Details? . . . Just Ask
Contact NATIONAL EXPOSITIONS
—or— return coupon

damentals for Non-Programmers." Contact: CTG (see September 10-12).

September 12-14 Digital Seminar Program, New York, NY: "The C Programming Language." Contact: Digital Educational Services (see September 5-7).

September 13 AT&T Technologies, Lisle, IL: "Shell Command Language for Users." Contact: AT&T (see September 5).

September 13-14 Computer Technology Group, San Francisco, CA: "Shell as a Command Language." Contact: CTG (see September 10-12).

September 17 NCR Education Seminars, Dallas, TX: "UNIX System Administration." Contact: NCR (see September 10).

September 17 NCR Education Seminars, Dayton, OH: "C Programming." Contact: NCR (see September 10).

September 17-19 Computer Technology Group, New York, NY & Washington, DC: "UNIX Fundamentals for Programmers." Contact: CTG (see September 10-12).

September 17-21 Bunker Ramo Information Systems, Trumbull, CT: "Intro to UNIX." Contact: Bunker Ramo (see September 10-14).

September 17-21 Computer Technology Group, San Francisco, CA: "C Language Programming." Contact: CTG (see September 10-12).

September 17-21 Plum Hall Training, Princeton, NJ: "C Programming Workshop." Contact: Plum Hall, 1 Spruce Ave., Cardiff, NJ 08232. 609/927-3770.

September 17-21 Uniq Digital Technologies Seminar, Chicago, IL: "UNIX System Administration." Contact: Uniq (see September 10-14).

September 18-21 Integrated Computer Systems Seminars, Washington, DC: "Hands-on UNIX Workshop." Contact: Integrated Computer Systems Seminars, 6305 Arizona Place, Los Angeles, CA 90045. 800/421-8166, or in CA, 800/352-8251.

September 19-21 CAPE Seminar, San Francisco, CA: "A User-Oriented Evaluation Three-Day Seminar: UNIX." Contact: CAPE (see September 5-7).

September 20-21 Computer Technology Group, New York, NY & Washington, DC: "Shell as a Command Language." Contact: CTG (see September 10-12).

September 24 AT&T Technologies, Sunnyvale, CA: "C Language for Experienced Programmers." Contact: AT&T (see September 5).

September 24 AT&T Technologies, Lisle, IL: "Software Development Under the UNIX System." Contact: AT&T (see September 5).

September 24 AT&T Technologies, Lisle, IL: "UNIX

System Device Drivers." Contact: AT&T (see September 5).

September 24 NCR Education Seminars, Chicago, IL: "C Programming." Contact: NCR (see September 10).

September 24 NCR Education Seminars, Dayton, OH: "UNIX System Administration." Contact: NCR (see September 10).

September 24-25 Computer Technology Group, San Francisco, CA: "Shell Programming." Contact: CTG (see September 10-12).

September 24-26 CAPE Seminar, Denver, CO: "A User-Oriented Evaluation Three-Day Seminar: UNIX." Contact: CAPE (see September 5-7).

September 24-28 Bunker Ramo Information Systems, Trumbull, CT: "C Programming." Contact: Bunker Ramo (see September 10-14).

September 24-28 Computer Technology Group, New York, NY & Washington, DC: "C Language Programming." Contact: CTG (see September 10-12).

September 24-28 Structured Methods Seminar, New York, NY: "C Programming Workshop." Contact: Structured Methods (see September 10-14).

September 24-28 Uniq Digital Technologies Seminar, Chicago, IL: "C Programming Language." Contact: Uniq (see September 10-14).

September 25 Computer Technology Group, Chicago, IL: "UNIX Overview." Contact: CTG (see September 10-12).

September 25-28 Integrated Computer Systems Seminars, Washington, DC: "Programming in C: Hands-on Workshop." Contact: ICSS (see September 18-21).

September 26 AT&T Technologies, Lisle, IL: "UNIX System Tools." Contact: AT&T (see September 5).

September 26-28 Computer Technology Group, Chicago, IL: "UNIX Fundamentals for Non-Programmers." Contact: CTG (see September 10-12).

September 26-28 Computer Technology Group, San Francisco, CA: "Using Advanced UNIX Commands." Contact: CTG (see September 10-12).

OCTOBER

October 1 AT&T Technologies, Sunnyvale, CA: "Shell Command Language for Programmers." Contact: AT&T (see September 5).

October 1 AT&T Technologies, Hopewell, NJ: "Fundamentals of the UNIX Operating System for Users." Contact: AT&T (see September 5).

October 1 AT&T Technologies, Lisle, IL: "Fundamentals of the UNIX Operating System for Programmers." Contact: AT&T (see September 5).

October 1 NCR Education Seminars, New York, NY: "UNIX Operating System." Contact: NCR (see September 10).

BEAT OUR BENCHMARK.



GET THIS WATCH.

Take the Plexus Challenge.

Everybody in the UNIX-based supermicro business talks performance.

Only Plexus dares to prove it... with the Plexus Challenge:

Show us a UNIX-based supermicro that can beat Plexus — running any recognized benchmark (or even your own application code*) — and we'll present you with this elegant Heuer chronograph valued at \$485.00.

How do we dare make this offer?

Simple: our multiprocessor architecture and highly-tuned version of UNIX leave the single-CPU competition in the dust. Multibus architecture and intelligent I/O processors let you expand modularly from



5 to 40 users — without changing a single byte of object code.

Are you thinking "Fine, but I really don't need all that performance"? Well, some of our customers tried other hardware first. They used to think the same thing.

Challenge details: This offer is extended to bona fide OEM's, value-added resellers, and volume end-users in the multiuser commercial UNIX systems market. Limit one award per company. Offer expires Jan. 31, 1985 unless withdrawn earlier. For complete rules and participation information, call (800) 556-1234 Ext. 560 (in Calif. (800) 441-2345 Ext. 560)

PLEXUS
Supermicros built for speed.

Plexus Computers Inc., 3833 North First Street, San Jose, CA 95134

ADVERTISERS INDEX

Accucom Data	16	Mark Williams	17
Aim Technology	21	Micro Focus	87
Applix	9	Microware Systems	71
AT&T Technologies	10	MIPS Software	94
BASIS	92	Network Consulting	90
Century Software	76	Network Research	99
Cincom	79	NIAL Systems	92
Codata	69	Oasys	89
Computer Cognition	24	Oregon Software	93
Computer Consoles, Inc.	48	Plexus	107
Computer Corp. of America	86	Prodata	92
Computer Faire	61	Productivity Products	84
Computer Methods	11	Quality Software	108
Computer Technology Group	67	Relational Database Systems	5, 6, 7
Convergent Technology	Center Spread	Relational Technology Inc.	65
COSI	19	Rhodnius	31
Cray Research	77	Santa Cruz Operation	Cover III
Emerald City Inc.	47	Scientific Placement	66
Faircom	70	SHA Computers	84
Gould	75	Soft Faire	97
Handle Corp.	15	Software Results	80
Heurikon	98	Southwind Software	95
Hewlett-Packard	50, 51	Tektronix	82
Human Computing Resources	13	Telecon	94
I.B.C.	Cover IV	Telos Consulting Services	96
Image Network	56, 57	Unify	Cover II, 1
ITS	103	Unipress Software	81, 83
JMI	41	Uniq Digital	73
Korsmeyer Electronic Design	96	Unisoft	45
Lachman Associates	91	UNIX EXPO	105
Lantech	85	User Training Corp.	66
Lawrence Livermore Labs	76	Visual Technology	58, 59
Logical Software	100	Xidak	49
Manx Software	25		

**LOTUS 1-2-3* MOVE OVER
QUALITY SOFTWARE
ANNOUNCES
Q-CALC (version 3.0)**

- spreadsheet
- data management
- forms processing
- graphics

PLUS
it runs on UNIX

For more information write/call
Quality Software Products
348 S. Clark Dr.
Beverly Hills, CA 90211
213-659-1560

*Lotus 1-2-3 is a trademark of Lotus Development Corp.
Circle No. 69 on Inquiry Card

BACK ISSUES AVAILABLE

- June/July 1983
- August/September 1983
- October/November 1983
- December/January 1984
- February/March 1984
- April/May 1984
- June 1984
- July 1984
- August 1984

All back issues are \$4.95, including postage and handling. Enclose payment or credit card information or call 206/271-9605.

Name _____
 Company _____
 Address _____
 City _____ State _____ Zip _____
 M/C or VISA _____
 Exp. Date _____

THE WORLD'S MOST POPULAR

UNIX SYSTEM

FOR THE

IBM PC

AND

LISA 2

Now, SCO brings the world's most popular version of AT&T's UNIX™ Operating System — and a comprehensive line of applications software — to the IBM® Personal Computer™ and the Apple® Lisa 2™. XENIX™ from SCO is simply the best microprocessor-based UNIX you can buy. **No one else in the UNIX business provides you with better support, and no other version of UNIX gives you all these features:**

VISUAL SHELL:

The XENIX visual shell provides you with a screen-oriented, menu-driven interface—with selectable color for the IBM PC.

MULTIUSER:

With XENIX, 3 or more users may efficiently use a single IBM PC or Lisa 2 simultaneously.

MOST APPLICATIONS:

With nearly 80% of all micro-based UNIX systems running XENIX, more applications are available for XENIX than for any other version of UNIX.

The powerful family of XENIX systems software and applications is available today from SCO — your UNIX Systems Software Company since 1978 — to set a new standard for personal computer productivity.

MULTISCREEN™:

SCO's Multiscreen™ allows you to run multiple programs simultaneously and view each program's screen with a touch of a key.

MOST COMPLETE:

XENIX is the most complete version of the UNIX System available for micros — including Microsoft's commercial enhancements and key extras from Berkeley 4.2 UNIX.

PC-DOS:

XENIX lets you read and write PC-DOS files on the IBM PC—and a cross-development environment will be available soon.

DOCUMENTATION:

SCO's XENIX documentation is written to meet the needs of the small systems marketplace — well-organized, clear, concise.

500 CHESTNUT STREET
P.O. BOX 1900
SANTA CRUZ, CALIFORNIA 95061

SCO
THE SANTA CRUZ OPERATION

YOUR UNIX SYSTEMS SOFTWARE COMPANY

408/425-7222

Let's go to work.

UNIX is a trademark of Bell Laboratories
IBM is a registered trademark and IBM PC, IBM Personal Computer are trademarks of International Business Machines Corp.
Apple is a registered trademark and Lisa 2 is a trademark of Apple Computer, Inc.
XENIX is a trademark of Microsoft Corp.
Multiscreen is a trademark of The Santa Cruz Operation, Inc.
© 1984 The Santa Cruz Operation, Inc.



UNIX™ HORSEPOWER!

There are a lot of UNIX based systems on the market today claiming to be "SUPERMICROS". But do they really have what it takes to run multi-user UNIX well? The IBC ENSIGN™ does and here's why:

FAST MEMORY: No computer running at **any** clock speed can run faster than it's overall memory design. The ENSIGN has up to 8MB of 120nsec memory with dual bit error correction. With IBC's proprietary memory management, **all** of this memory runs with no wait states as fast as the 68000 CPU will go. Compare this to other systems running only small cache memories at full speed. Other multiple user systems cannot load all their programs into a small cache memory. Their systems slow down considerably under a heavy multi-user load.

INTELLIGENT SERIAL I/O CONTROLLER: Even the fastest CPU will slow down when it's trying to handle interruptions from multiple on-line users. The ENSIGN provides slave serial I/O CPU's **and** FIFO buffering for both input and output. The result is the ENSIGN's ability to support up to 32 users, with heavy serial I/O demand, while leaving the main 68000 CPU free to run with little serial I/O overhead.

INTELLIGENT DISK CONTROLLER AND HIGH PERFORMANCE DISK DRIVES: The ENSIGN has a slave CPU to handle all disk operations, **plus** 16K of disk buffering. IBC's proprietary disk DMA allows high speed data transfer to main memory without slowing down the main CPU. Further, the ENSIGN

supports SMD type 8" hard disks with much faster seek times and transfer rates than 5¼" hard disks usually found in personal desk top computers.

THE RESULTS: The IBC ENSIGN runs multi-user UNIX at performance levels not attainable by other supermicros.

Call IBC and get a copy of IBC's multi-user benchmarks—benchmarks that test 8 users running large CPU programs, with heavy disk I/O **and** heavy serial I/O simultaneously. You'll find that nothing can compare to the ENSIGN.



If you want to run multi-user UNIX on a high performance system with up to 32 users, 8MB memory, and over 1,000MB disk storage, see the IBC ENSIGN.

Outside The USA

IBC / Integrated Business Computers

21621 Nordhoff Street
Chatsworth, CA 91311
(818) 882-9007
Telex No. 215349

Within The USA

IBC / DISTRIBUTION

1140 36th Street Suite 212
Ogden, Utah 84403
(801) 621-2294

UNIX is a trademark of Bell Laboratories

Circle No. 67 on Inquiry Card